A PRIMERA REVISTA DE PROGRAMACIÓN EN CASTELLANO

SOLO PROGRAMADORES

) IV, NÚMERO 45

975 Ptas.

'ENDER PETICIONES DE PÁGINAS HTML

REACIÓN DE UN SERVIDOR WEB CON DELPHI

CONTENIDO DEL CD-ROM

- PowerBuilder 6.0
- SQL Anywhere Prof.
- TAPI 2.1
- PowerSite
- Apache
- Ports FreeBSD
- Xemacs

JAVA SERVLETS
Primer análisis de la estructura básica

WINDOWS NT SERVER
Seguridad en el servidor con SSL

VISUAL BASIC
Controles ActiveX avanzados

SIMULACIÓN DE SISTEMAS
Práctica con sistema continuo

C++
Borland C++ Builder 3

HARDWARE
El control MSComm de Visual Basic

LINUX
Aplicaciones X-Window con Motif

COBOL
MicroFocus NetExpress

8 413042 303299

SÓLO PROGRAMADORES

Número 45
SÓLO PROGRAMADORES
es una publicación de
TOWER COMMUNICATIONS

Director Editor
Antonio M. Ferrer Abelló
aferrer@towercom.es
Coordinador Técnico
Eduardo de Riquer

Colaboradores
Enrique Díaz, José Antonio Mendoza, Ernesto Schmitz, Chema Alvarez, Carlos Álvaro, Constantino Sánchez Ballesteros, Arturo Álvaro, Jorge F. Delgado, Xesco Hernández,Juan Carlos Sastre, Enrique de la Lastra, Jordi Agost

Maquetación
Susana Llano
Tratamiento de Imagen
María Arce Giménez

Publicidad
Erika de la Riva (Madrid)
Tel.: (91) 661 42 11
Pepín Gallardo (Barcelona)
Tel.: (93) 213 42 29

Suscripciones
Begoña Alvarez
Tel. (91) 661 42 11 Fax: (91) 661 43 86
suscrip@towercom.es

Laboratorio
Óscar Rodríguez (jefe)
oscarrf@towercom.es
Javier Amado
jamado@towercom.es
Seryicio Técnico
Oscar Casado
ocasado@towercom.es

Filmación
Megatipo
Impresión
Gráficas Reunidas
Distribución
SGEL
Distribución en Argentina / Chile / Colombia / México / Venezuela
Capital: Huesca y Sanabria
Interior: D.G.P.

TOWER COMMUNICATIONS
Director General
Antonio M. Ferrer Abelló
Director Financiero
Francisco García Díaz de Liaño
Director de Producción
Carlos Peropadre
Directora Comercial
Carmina Ferrer
carmina@towercom.es

Redacción, Publicidad y Administración
C/ Aragoneses, 7
28108 Pol. Ind. Alcobendas (MADRID)
Telf.: (91) 661 42 11 / Fax: (91) 661 43 86

La revista Sólo Programadores no tiene por qué estar de acuerdo con las opiniones escritas por sus colaboradores en los artículos firmados. El editor prohibe expresamente la reproducción total o parcial de los contenidos de la revista sin su autorización escrita.

Depósito legal: M-26827-1994
ISSN: 1134-4792
PRINTED IN SPAIN
COPYRIGHT 31-07-98

EDITORIAL

Compuestos y sin código

Aunque no todos los lectores compartan la opinión del número anterior sobre las dualidades y encrucijadas de la programación, sea cual sea el punto de vista de cada uno, es indudable que si hay algún tema que nadie debe descuidar es el relacionado con Internet. La verdad es que no hace falta ser un virtuoso adivino para percartarse de que la única apuesta segura en estos momentos pasa por la reconducción de nuestras técnicas de programación hacia este fenómeno. El futuro de todos los campos de la Informática pasa por alcanzar ese ansiado estatus de comunicación total, gracias al que dicho sea de paso, se ha disparado la demanda de profesionales vinculados con este sector. Dentro de un mundo tan exigente, (donde parece que los ordenadores son los que se sientan delante de ti y experimentan sobre tus posibilidades), resulta alentador partir de una base sólida desde la que realizar un planteamiento con muchas probabilidades de éxito.

Una de las apuestas que viene al caso, por su relación con Internet y con las cualidades innatas de los ordenadores para dejarnos en ridículo en el peor momento (que se lo digan al padre de Windows 98), se basa en el uso de componentes. Se podrían describir como las cajas negras del software, en las que el programador solamente debe aprender la forma de utilizarlas sin necesidad de comprender su implementación interna. Por tanto, actúan como pasarela entre las herramientas de programación y los objetivos planteados por los clientes. Los más veteranos pensarán que no se trata de un tema nuevo, puesto que siempre han existido de una forma más o menos directa (unidades, paquetes, librerías, objetos, etc.), pero las facilidades de comunicación actuales permiten acceder a un mercado inmenso sin salir de casa, junto con la posibilidad de estar actualizado constantemente.

Parece que no hay escapatoria, y que la programación pasa indefectiblemente por la reutilización de código. Ante un planteamiento similar parece que nos volveremos más que prescindibles. En realidad esto último es difícil que se produzca (eso esperamos todos), puesto que los programadores siempre tendremos que situar nuestras habilidades por encima de la media, es decir, nuestros esfuerzos se orientarán al desarrollo de componentes muy específicos para terceros. Desde la revista esperamos que los artículos consigan aumentar el nivel de todos nosotros, con el fin de no ceder ante ninguna tendencia y permanecer de una forma u otra, siempre en la cresta de la ola.

Tenéis a vuestra disposición la dirección de correo electrónico de la revista: solop@towercom.es, donde podéis enviar vuestras sugerencias sobre los contenidos, los temas que queráis que incluyamos y los productos que os gustaría que distribuyéramos con el CD-ROM, que todos los meses regalamos. Sois nuestros mejores colaboradores.

solo PROGRAMADORES

45

6 Noticias
LO ULTIMO EN DESARROLLO

En esta sección podemos encontrar las últimas novedades en software, así como los detalles de las alianzas en materia de desarrollo. Como todos los meses, lo último de lo último.

14 Programación Hardware
CONTROL DEL MÓDEM CON VISUAL BASIC (II)

Este mes utilizaremos el control MSComm para realizar un programa que podrá efectuar llamadas de teléfono desde nuestro ordenador a cualquier persona del mundo y plantear conversaciones on-line. El funcionamiento del desarrollo será muy simple pero a la vez muy útil.

17 Windows NT Server
INTERNET INFORMATION SERVER (II): SEGURIDAD CON SSL

El articulo de este mes profundiza en aspectos avanzados de configuración así como en el polémico asunto de la seguridad en los servidores de red.

23 DirectX
PROGRAMACIÓN DE DIRECTX CON DELPHI (V)

Los detalles internos de los ficheros que han sido creados con la librería de imágenes es el tema principal del artículo de este apartado. En él se definirá el FX Editor, describiendo por otro lado la programación de arcades algunos arcades de carreras.

44 Programación de Sockets
CREACIÓN DE UN SERVIDOR WEB CON DELPHI

En nuestro artículo de portada de este mes intentamos profundizar en el tema Internet y más concretamente en la creación de un programa servidor Web. Para ello utilizaremos uno de los lenguajes con mayor proyección de los últimos tiempos y que a buen seguro hará las delicias de todos los desarrolladores, se trata de Delphi.

38 Simulación de sistemas
PRACTICA CON UN SISTEMA CONTINUO

Tras la introducción teórica realizada en artículo anterior con éste comenzmlos la parte práctica, simulando la amortiguación de un coche.

54 Linux
PROGRAMACIÓN DE APLICACIONES X-WINDOW CON MOTIF

El objetivo fundamental del artículo de este mes es el de introducir al desarrollador de C en la programación de aplicaciones utilizando librerías Motif. Para esto necesitaremos algunos conocimientos básicos que encontraremos muy útiles.

57 Java
SERVLETS (II): LA CARA OCULTA DE LA WEB

En esta segunda entrega del curso sobre Java se van a completar los pasos prácticos necesarios para hacer funcionar sin problemas un Servlet y se podrá realizar un primer análisis de la estructura básica y de su composición típica.

61 Herramientas de programación
MICROFOCUS NETEXPRESS COBOL

Este mes analizamos una de las herramientas que más nos ha sorprendido últimamente. Se trata de un entorno que permite reutilizar código antiguo escrito en Cobol y actualizarlo automáticamente para su uso y ejecución en entornos Internet e intranet, que tan de moda están últimamente.

70 Libros
PARA TODOS LOS GUSTOS Y NECESIDADES

Ante las sugerencias recibidas, este mes aumentamos las páginas que ocupa esta sección. En ella pueden encontrarse manuales que servirán como ayuda para solucionar nuestras dudas a la hora de crear nuevas aplicaciones o de modificar las que tenemos.

64 Herramientas de programación
C++ BUILDER 3. A LA TERCERA VA LA VENCIDA

Es la evolución natural de las herramientas de desarrollo en C++.
Para ello conjuga la rapidez de la programación visual con la potencia del propio lenguaje.
La nueva versión está pensada para resolver las necesidades de todo tipo de desarrolladores de sistemas.

72 CORREO DEL LECTOR

En esta sección se pretende dar respuesta a todas las dudas que puedan tener nuestros lectores, pudiendo dirigirse a los propios autores o bien a la dirección de correo de nuestra publicación: solop@towercom.es.

76 CONTENIDO DEL CD-ROM

De nuevo volvemos a volcarnos sobre Linux. En esta ocasión incluimos de nuevo un segundo CD-ROM que hará las delicias de los desarrolladores más audaces y de aquellos que nos piden habitualmentè una versión en castellano. Además de esto nuestro CD-ROM habitual incluye, como todos los meses, las últimas novedades en el mercado del desarrollo.

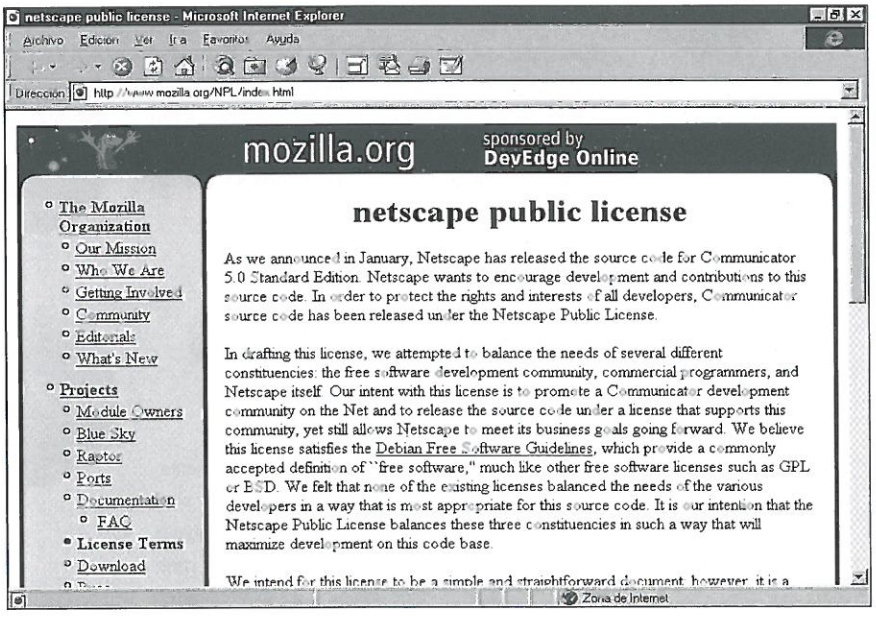
Noticias

El site ha sido todo un éxito para los desarrolladores

NETSCAPE PUBLICA EN MOZILLA.ORG EL NUEVO CÓDIGO FUENTE DE COMMUNICATOR 5.0

Netscape ha anunciado la publicación del nuevo código fuente de Communicator 5.0 en su versión para desarrolladores, que puede descargarse del sitio web **www.mozilla.org**. Con este nuevo código se incluye un analizador de XML avanzado y una versión preliminar de una sistema de creación de diseños de próxima generación. Los desarrolladores de contenidos para Internet que deseen programar con XML ya pueden hacerlo utilizando el software cliente de nueva generación de Netscape. Además la inclusión por parte de Mozilla del analizador XML representa la primera aportación importante al código fuente por parte de desarrolladores independientes desde la publicación del código anterior hace un mes aproximadamente.

Se presenta una mejora en la arquitectura debido al innovador sistema auxiliar para el diseño en programación que permitirá conseguir prestaciones superiores. Este sistema de creación de diseños acabará siendo totalmente compatible con HTML 4.0, el DOM de W3C, las hojas de estilo en cascada, XML y RDF. El sitio web está patrocinado por DevEdge Online y es un canal desde el que Netscape ofrece recursos técnicos para los desarrolladores así como información para la comercialización de aplicaciones Webs, intranets y extranets basadas en la plataforma de Netscape. Más información en **http://home.netscape.com/communicator** y en **http://developer.netscape.com**


netscape public license - Microsoft Internet Explorer
Archivo Edición Ver Ir a Favoritos Ayuda
Dirección http://www.mozilla.org/NPL/index.html
mozilla.org
sponsored by DevEdge Online
The Mozilla Organization
Our Mission
Who We Are
Getting Involved
Community
Editorials
What's New
Projects
Module Owners
Blue Sky
Raptor
Ports
Documentation
FAQ
License Terms
Download
netscape public license
As we announced in January, Netscape has released the source code for Communicator 5.0 Standard Edition. Netscape wants to encourage development and contributions to this source code. In order to protect the rights and interests of all developers, Communicator source code has been released under the Netscape Public License.
In drafting this license, we attempted to balance the needs of several different constituencies: the free software development community, commercial programmers, and Netscape itself. Our intent with this license is to promote a Communicator development community on the Net and to release the source code under a license that supports this community, yet still allows Netscape to meet its business goals going forward. We believe this license satisfies the Debian Free Software Guidelines, which provide a commonly accepted definition of "free software," much like other free software licenses such as GPL or BSD. We felt that none of the existing licenses balanced the needs of the various developers in a way that is most appropriate for this source code. It is our intention that the Netscape Public License balances these three constituencies in such a way that will maximize development on this code base.
We intend for this license to be a simple and straightforward document, however, it is a
Zona de Internet

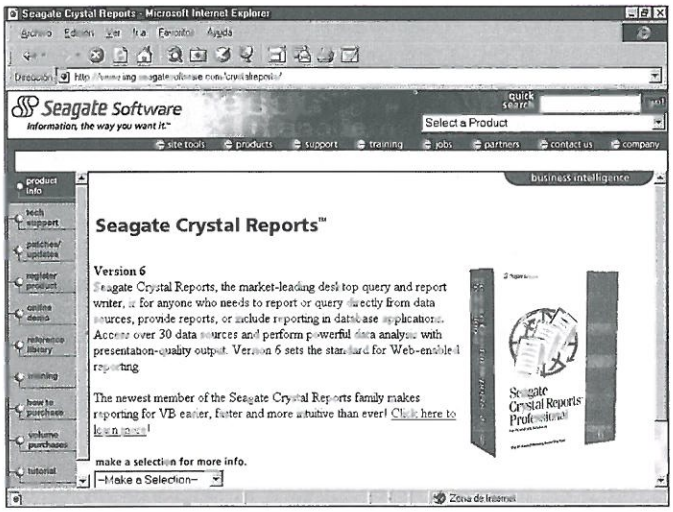
Crystal Reports integrará informes en Java
SEAGATE SOFTWARE OFRECE SOPORTE A LA NUEVA VERSIÓN BETA MICROSOFT VISUAL J++

Seagate Software ha anunciado que soportará la versión beta de la tecnología Microsoft Visual J++ 6.0, el conjunto de herramientas de desarrollo de Windows para programadores que construyen applets y aplicaciones compatibles con Java.

El software de creación de informes y consultas para sistemas de sobremesa líder en la industria, Seagate Crystal Reports, utiliza unas potentes características de desarrollo tales como Crystal Report Engine Automation Server ofreciendo así a los desarrolladores de aplicaciones Web y Windows, una creación de informes superior. Debido a su diseño modular y a una distribución del tiempo de ejecución gratuito, Crystal Report Engine se puede utilizar como herramienta RAD para integrar la creación de informes en aplicaciones Visual J++.

Tiene capacidad para acceder a más de 30 fuentes de datos, potencia para analizar datos, opciones para distintos tipos de informes, presentaciones de calidad y flexibilidad en la integración de creación de informes en las aplicaciones.

Ofrece además la misma potencia y facilidad de uso en la creación de informes Web, lo que permite un rápido desarrollo de informes a cualquier servidor Web compatible con ISAPI/NSAPI, así como también con Active Server Pages para las aplicaciones desarrolladas para la Web.


Seagate Crystal Reports - Microsoft Internet Explorer
Seagate Software
Information, the way you want it.
Select a Product
site tools
products
support
training
jobs
partners
contact us
company
business intelligence
Seagate Crystal Reports™
Version 6
Seagate Crystal Reports, the market-leading desktop query and report writer, is for anyone who needs to report or query directly from data sources, provide reports, or include reporting in database applications. Access over 30 data sources and perform powerful data analysis with presentation-quality output. Version 6 sets the standard for Web-enabled reporting.
The newest member of the Seagate Crystal Reports family makes reporting for VB easier, faster and more intuitive than ever! Click here to learn more!
make a selection for more info.
-Make a Selection-
Seagate Crystal Reports Professional

Si desea obtener más información sobre los productos de Seagate Software puede consultar la dirección **www.seagatesoftware.com**

Con WebFocus Suite de Informations Builders
APPLETS JAVA 100%

La empresa Information Builders ha anunciado el lanzamiento de WebFOCUS Suite, un conjunto de applets 100% Java que amplían de forma significativa la funcionalidad de WebFOCUS Application Server.

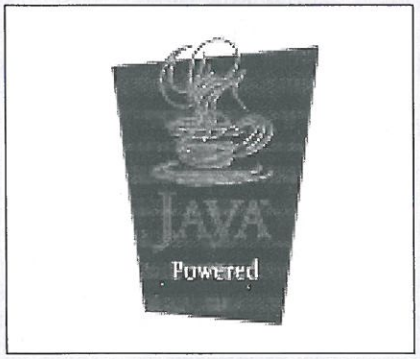
Además proporciona capacidades de edición electrónica y generación de informes dinámicos a través de la Web, desde más de 70 tipos de bases de datos diferentes.

Como respuesta a la demanda de los usuarios de robustas herramientas basadas en Java para la generación de informes de usuario final, la nueva familia de productos WebFOCUS Suite añade capacidades para crear informes gestionados, navegación OLAP, así como amplias capacidades de planificación y distribución de informes.

Por otro lado el entorno WebFOCUS Managed Reporting permite a sus usuarios con poca o ninguna experiencia en bases de datos crear sofisticados informes utilizando objetos que previamente han sido construidos por un administrador de bases de datos con una mayor exactitud y eficacia.

Los objetos pueden ser utilizados para crear una amplia variedad de consultas ad hoc e informes diseñados para el usuario. El applet no requiere otro software adicional en el ordenador de sobremesa más que un navegador Web estándar, como Netscape Navigator o Internet Explorer.

Esta Suite incluye además un broker de informes, una utilidad llamada WebFOCUS Report Broker que es un motor inteligente basado en Java que centraliza la planificación y distribución de informes a través de la Web, E-mail, y dispositivos de comunicaciones como impresoras y faxes.



UnixWare 7.0
POTENCIA UNIX CON VERSATILIDAD INTEL

SCO anuncia la disponibilidad de UnixWare 7, adelantando así la salida del sistema operativo más avanzado de la industria para servidores Intel de gran volumen.

La compañía desveló los detalles de la línea de productos UnixWare 7, que se presenta en cinco configuraciones de propósito específico.

Este sistema operativo combina la potencia, fiabilidad y escalabilidad del Sistema UNIX con el valor y la versatilidad de la plataforma Intel.

La línea de productos UnixWare 7 está equipada con niveles de memoria y sistema de ficheros sin precedentes para servidores Intel, así como extensiones de 64 bits, preparándose para los futuros sistemas Intel de 64 bits del año próximo.

Las configuraciones que soporta UnixWare 7 son las siguientes: Enterprise Edition, Departmental Edition, Intranet Edition, Messagin Edition y Base Edition, de todo ello puede encontrar más información en la página Web **www.sco.es**

MS Foundation Classes
LIBRERÍAS PARA DESARROLLADORES

La compañía Microsoft ha anunciado el lanzamiento de las Windows Foundation Classes, un conjunto de librerías de clases para Java que permiten a los desarrolladores construir en Java potentes aplicaciones nativas para Windows.

También proporciona un entorno de trabajo orientado a objetos que unifica el desarrollo para las APIs de Windows de 32 bits con el desarrollo Web mediante un sencillo modelo de programación que abarca tanto el servidor como el cliente. Al mismo tiempo que Microsoft anuncia este lanzamiento, varios suministradores y desarrolladores han anunciado sus planes para soportar las Windows Foundation Classes.

Las WFC combinan los beneficios de productividad del lenguaje Java con las capacidades naturales de la plataforma Windows, consiguiendo así una mayor facilidad de creación de un software más sofisticado y útil basado en Windows. Además también proporciona una potente encapsulación orientada a objetos de Win32, así como HTML/HTML dinámico, de modo que los desarrolladores dispongan de un modelo consistente para la construcción de aplicaciones que puedan funcionar stand-alone, en un browser en el cliente o en el lado del servidor.

Windows Foundation Classes incluye también un poderoso conjunto de clases de acceso de datos para la construcción de aplicaciones cliente/servidor basadas en datos, con acceso tanto a datos estructurados y con soporte para usuarios móviles.

Ya está disponible una versión preliminar de las Windows Foundation Classes como parte de Visual J++ 6.0 Technology Preview 1. Para más información se puede consultar en la página **www.microsoft.com/java/**

Alliance Generation y CT Access
LOS PRODUCTOS DE NATURAL MICROSYSTEMS SOPORTAN SOLARIS DE SUN

Natural MicroSystems anuncia que sus líneas de productos Alliance Generation y CT Access soportarán el entorno operativo de Solaris de Sun en los sistemas Ultra basados en PCI de Sun así como en los sistemas basados en Intel.

La versión CT Access 2.0 ya proporciona soporte para Solaris 2.6 para los procesadores Intel y el soporte para los procesadores SPARC se halla actualmente en la fase de prueba beta.

Natural M. se ha convertido en proveedor líder de plataformas abiertas integradas en la infraestructura de red PSTN e ISP gracias a la experiencia y las capacidades multiplataforma de Sun. Además soportando Solaris se puede abrir un mercado potencial e importante para los socios de la compañía.

CT Access es un API unificado de la casa Natural Microsystems que proporciona el interfaz de programación de aplicaciones que expone las capacidades de la línea de productos interfaces de red y tarjetas de recursos SP Alliance Generation de Natural MicroSystems, para procesamiento de llamadas y medios.

La dirección internet en la que puede consultar es **www.nmss.com**

Para los ISVs basados en Unix y estaciones de trabajo

MICROSOFT E INTEL ANUNCIAN UN NUEVO PROGRAMA DE MIGRACIÓN

El nuevo programa de migración MAP, desarrollado por Microsoft e Intel ayudará a los ISVs basados en UNIX a construir aplicaciones de estaciones de trabajo para la plataforma Windows NT e Intel.

MAP ofrecerá soporte directo para seleccionar a los ISVs en los mercados clave y con esa actividad permitirles capitalizar la integración de negocios, innovaciones y bajo coste de propiedad, algo que ha hecho de Windows NT y la arquitectura Intel la combinación de tecnologías preferida para la informática corporativa.

Para acelerar la migración de soluciones UNIX, Microsoft e Intel trabajarán con el equipo técnico de los fabricantes de sistemas de estaciones de trabajo, con el objetivo de ofrecer un programa con descuentos en estaciones de trabajo basadas en el procesador Pentium II.

Además ofrecerán una amplia variedad de servicios directos a las empresas participantes en el programa, como formación para desarrolladores on-site, revisiones de la arquitectura de la aplicación y optimización y desarrollo de nuevas aplicaciones.

Puede ampliar información en: **www.intel.com** y **www.microsoft.com/spain**

Según información difundida en el pasado CEBIT 98

MÁS DE 300 SOLUCIONES INDUSTRIALES UTILIZAN LA PLATAFORMA TECNOLÓGICA DE MICROSOFT

Durante la pasada edición de CeBIT'98, Microsoft demostró el éxito de su plataforma BackOffice en un amplio abanico de sectores industriales, con la exhibición de más de 300 soluciones profesionales desarrolladas sobre los productos servidores de Microsoft. Los suministradores de software se benefician de la flexibilidad, escalabilidad y facilidad de uso de las tecnologías de servidor Microsoft como plataforma idónea para soluciones profesionales.

Desde la creación de la división ADCU de Microsoft, ésta ha trabajado con proveedores de software de todo el mundo, líderes en diferentes sectores.

Productos innovadores para contabilidad, gestión de documentos y clientes, comercio electrónico, mercados financieros, etc. vienen a demostrar la relevancia de las plataformas Microsoft en esos sectores industriales. Más información en **www.microsoft.com/spain**

Integración de redes

ENTRE EL S.O WINDOWS Y LOS MAINFRAMES

Con la intención de facilitar a las empresas la integración de aplicaciones basadas en el sistema operativo Windows y los mainframes compatibles de IBM, las compañías Cisco Systems y Microsoft han presentado un nuevo programa para ayudar a la PYME.

El trabajo en conjunto de estas dos empresas, permitirá a las demás compañías preservar la inversión realizada en los mainframes instalados en la actualidad, a la vez que podrán beneficiarse de las nuevas aplicaciones construidas sobre la plataforma Windows y de las nuevas tecnologías de soluciones de red que está aportando Microsoft en este campo.

En este sentido la combinación de los routers de Cisco y de Microsoft SNA Server, forman una solución potente de alta velocidad.

Para conseguir con éxito la implantación de esta combinación de productos ambas compañías llevarán a cabo una serie de pruebas conjuntas de interoperabilidad y rendimiento, poniendo a su vez a disposición de los clientes una guía de configuración y optimización para que puedan analizarse las necesidades y aportaciones que puede hacer al respecto el nuevo proyecto. Más información en la página www.cisco.com o en www.microsoft.com.

PROGRAMACIÓN HARDWARE

Control del módem con Visual Basic (II)

Constantino Sánchez Ballesteros (Constantino@redestb.es)

En este artículo se aborda un programa, utilizando el control MSComm.
Con él se podrán efectuar llamadas por teléfono desde nuestro ordenador a cualquier persona del mundo. Gracias a este programa se podrán realizar conversaciones On-Line de todo tipo.

Antes de conocer los aspectos concretos que se refieren a este tema de programación hardware es necesario tener en cuenta dos detalles fundamentales:

- Si el lector no posee un módem con soporte de voz, debéis usar el telefonillo o microteléfono para poder hablar con la otra persona. Los modems que soportan voz (comúnmente llamados Voice-Modems), permiten usar el micrófono y altavoces instalados en el ordenador para poder hablar y oír, respectivamente.

- El teléfono y el módem deben estar conectados a la misma línea.

El funcionamiento del programa es realmente simple; tan sólo necesitamos pulsar el botón **Marcar** del formulario y a continuación tendremos que introducir el número de teléfono al que vamos a llamar. Seguidamente pulsaremos **Aceptar** y el módem comenzará a marcar el número seleccionado.

OPTION EXPLICIT

DefInt se encarga de establecer el tipo de datos predeterminado de variables y funciones cuyos nombres comienzan con caracteres especificados. En este caso, todas las variables cuyos nombres comiencen con cualquier carácter desde la A hasta la Z, serán convertidos al tipo *Integer*. El tipo de datos predeterminado sólo se puede anular mediante una asignación explícita con la instrucción **Dim**. Las instrucciones **Def***tipo* sólo se pueden utilizar a nivel de módulo (es decir, no se admiten dentro de procedimientos).

DefInt A-Z

- **.....Dim CancelFlag**

Este indicador (**CancelFlag**) se establece cuando el usuario elige *Cancelar*, es decir, que el usuario corta repentinamente la llamada. En conjunción con esta variable, el siguiente procedimiento es el encargado de activar o desactivar la llamada. La indicación de corte en la llamada será enviada al procedimiento **Dial**.

```
Private Sub CancelButton_Click()
   CancelFlag = True
   CancelButton.Enabled = False
End Sub
```

El procedimiento **Dial** es el encargado de efectuar el marcado del número de teléfono establecido por nosotros. Dicho número será pasado a la variable **Number$**. **AT** es el comando compatible Hayes *ATTENTION*, y es necesario para enviar comandos al módem. **DT** significa

"marcación por tonos" y es el utilizado por defecto en el programa.

Si se tienen problemas en este tipo de marcado, podéis utilizar el marcado por impulsos. El comando **Dial** utiliza tonos, en lugar de pulsos; para usar pulsos debéis cambiar **DT** por **DP**. El punto y coma indica al módem que vuelva al modo de comandos después de marcar.

NOTA: *Es necesario un retorno de carro, "vbcr", al enviar comandos al módem.*

```
Private Sub Dial(Number$)
  Dim DialString$, FromModem$, dummy
  DialString$ = "ATDT" + Number$ + ";" +
    vbCr
```

- **.....Dialstring$**

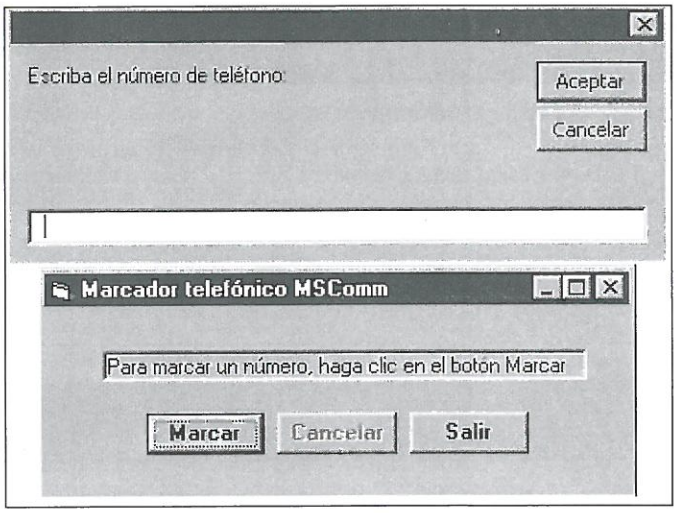
Almacena la cadena de comandos para efectuar el marcado. El siguiente paso consiste en configurar el puerto de comunicaciones. Suponiendo que el COM1 está siendo utilizado (normalmente por el ratón), estableceremos el *CommPort* al valor de 2, para usar el puerto serie COM2. Si vuestro módem está instalado en un COM diferente, sólo tenéis que cambiar el valor numérico al puerto correspondiente.

El comando Dial utiliza tonos, en lugar de pulsos; para usar pulsos se debe cambiar DT por DP

Nota: *Con algunos modems internos puede haber problemas al seleccionar su puerto COM, por lo tanto, si no se detecta en el puerto instalado en Windows, debemos probar con otros.*

- **....MSComm1.CommPort = 2**

Como vimos en el número anterior, debemos establecer los *baudios* para configurar la velocidad de comunicación. La propiedad *Settings* permite especificar la velocidad en *baudios*, la paridad y el número de *bits* de datos y parada. De forma predeterminada, la velocidad en *baudios* es 9600. La paridad sirve para la validación de los datos. Normalmente, no se utiliza y se establece a "N". El valor de *bits* de datos indica el número de *bits* que representa un bloque de datos. El *bit* de parada indica cuándo se ha recibido un bloque de datos.



Detalle de los formularios utilizados en el marcador telefónico.

- **....MSComm1.Settings = "9600,N,8,1"**

El siguiente paso corresponde a la apertura del puerto de comunicaciones. Esto debe hacerse asignando *true* al método **PortOpen** del control *MSComm1* utilizado por nosotros. Lógicamente, si no se abre, no podremos efectuar salida de datos al exterior. Mediante el tratamiento de errores (*On error*) nos aseguramos que el puerto que se va a utilizar cuando se abra esté disponible. La variable **Err** contendrá la activación de errores si ocurre alguno, en cuyo caso mostraríamos por pantalla un mensaje indicando el error devuelto:

```
On Error Resume Next
MSComm1.PortOpen = True
If Err Then
MsgBox "COM2 no disponible. Utilice otro
    puerto para establecer las comunicaciones."
Exit Sub
End If
```

Volcamos el buffer de entrada para realizar su vaciado mediante el valor "0".

- **...MSComm1.InBufferCount = 0**

Marcamos el número correspondiente utilizando la cadena establecida en la variable **DialString$**.

- **MSComm1.Output = DialString$**

En la siguiente rutina, esperamos a que el módem envíe el valor de retorno *OK* para asegurarnos de la buena progresión del marcado. La función *DoEvents* pasa el control al sistema operativo. De este modo, permitimos que siga activada la multitarea en Windows 95. El control no se devuelve hasta que el sistema operativo haya terminado de procesar los eventos en cola y se hayan enviado todas las teclas de la cola *SendKeys*.

El tipo de datos predeterminado sólo se puede anular mediante una asignación explícita con la instrucción Dim

Nota: *Cuando demos preferencia al procesador temporalmente dentro de un*

Tabla 1.- Constantes de protocolos

Constante	Valor	Descripción
comNone	0	Sin protocolo.
comXonXoff	1	Protocolo XON/XOFF.
comRTS	2	Protocolo RTS/CTS (Petición de envío/Preparado para enviar).
comRTSXOnXOff	3	Ambos protocolos (RTS y XON/XOFF).

Tabla 2.- Constantes OnComm

Constante	Valor	Descripción
comEvSend	1	Evento Send.
comEvReceive	2	Evento Receive.
comEvCTS	3	Cambio en la línea Preparado para enviar (CTS).
comEvDSR	4	Cambio en la línea Equipo de datos preparado (DSR).
comEvCD	5	Cambio en la línea Detección de portadora (CD).
ComEvRing	6	Detección de llamada.
comEvEOF	7	Fin de archivo.

Tabla 3.- Constantes de errores

Constante	Valor	Descripción
comEventBreak	1001	Recibida señal de interrupción
comEventCTSTO	1002	Tiempo de espera de Preparado para enviar
comEventDSRTO	1003	Tiempo de espera de Equipo de datos preparado
comEventFrame	1004	Error de trama
comEventOverrun	1006	Pérdida de información en puerto
comEventCDTO	1007	Tiempo de espera de Detección de portadora
comEventRxOver	1008	Desbordamiento del buffer de recepción
comEventRxParity	1009	Error de paridad
comEventTxFull	1010	Buffer de transmisión lleno
comEventDCB	1011	Error inesperado al recuperar Bloque de control de dispositivos (DCB) para el puerto

procedimiento de evento, debemos asegurarnos de que el procedimiento no se ejecute de nuevo desde una parte diferente del código antes de que devuelva la primera llamada. Esto podría causar resultados impredecibles. Además, no debemos utilizar la función DoEvents si existe la posibilidad de que otras aplicaciones interactúen con el procedimiento, de manera imprevista, durante el tiempo en que éste ha cedido el control.

```
Do
  dummy = DoEvents()
```

Si hay datos en el buffer, los lee:

```
If MSComm1.InBufferCount Then
  FromModem$ = FromModem$ +
  MSComm1.Input
```

Comprueba si hay "OK":

```
If InStr(FromModem$, "Aceptar") Then
```

Indica al usuario que descuelgue el teléfono.

```
    Beep
    MsgBox "Escriba el número y presione
  Intro o haga clic en Aceptar"
    Exit Do
  End If
End If
```

¿El usuario ha elegido Cancelar?

```
If CancelFlag Then
  CancelFlag = False
  Exit Do
End If
Loop
```

Desconectamos el módem mediante la propiedad *Output*. Para ello debemos mandar una cadena de comandos en la que se exprese esta acción. Mediante el comando compatible *Hayes* **ATH** logramos que el módem se desconecte.

```
MSComm1.Output = "ATH" + vbCr
```

Cerramos el puerto.

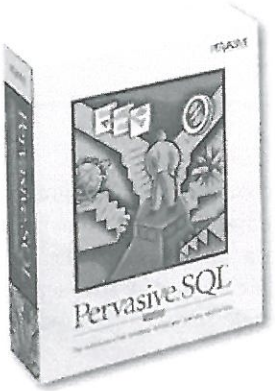
```
  MSComm1.PortOpen = False
End Sub
```

Mediante el comando compatible Hayes ATH logramos que el módem se desconecte

El siguiente procedimiento es el encargado de visualizar la ventana en la que se introducirá el número a marcar. Para dar salida a este nuevo formulario se utiliza la función **Inputbox**.

```
Private Sub DialButton_Click()
  Dim Number$, Temp$

  DialButton.Enabled = False
  QuitButton.Enabled = False
  CancelButton.Enabled = True
```

Obtiene el número a marcar.



Nuevo Pervasive SQL 7

El sistema
Cliente/Servidor
creado para que las **PYMES** Acierten

El sucesor de Btrieve

Pervasive SQL le ofrece lo mejor de ambos mundos:
Las mejores características de una arquitectura navegacional emparejada con la potencia de las capacidades relacionales.
Con esta estructura única, Pervasive SQL ofrece su imbatible gestión de transacciones y el máximo control en programación en adición a sus excepcionalmente rápidas consultas y manipulaciones de datos.
Pervasive SQL le ofrece lo último en operaciones confiables, sin mantenimiento, eficientes en recursos, con un rendimiento y fiabilidad de primera clase.
Pervasive SQL es la solución completa para gestionar bases de datos, con la gama más amplia de características para cubrir sus necesidades actuales y futuras. Piense si el sistema que busca o si su sistema actual le ofrece tantas ventajas como Pervasive SQL :

- ✔ Base de datos de 64 Gb.
- ✔ Motor multihilo de 32 bits.
- ✔ Operaciones continuas.
- ✔ Algoritmo de indexación hashing.
- ✔ Bloqueo a nivel de registros.
- ✔ Transacciones duraderas
- ✔ Balanceo automático de índices.
- ✔ Navegación bidireccional.
- ✔ Cache configurable.
- ✔ Integridad referencial.
- ✔ Optimización basada en costes
- ✔ ODBC
- ✔ Diccionario de datos
- ✔ Registro de archivos de Bitácora.
- ✔ Rehacer (Roll Forward).
- ✔ Expansión dinámica de ficheros.
- ✔ Utilidades.
- ✔ Manejo de licencias.

CARACTERISTICAS	Pervasive SQL 7	Su sistema de Base de Datos
Acceso Relacional y Transaccional	Si	¿?
Operaciones Libres de Mantenimiento	Si	¿?
Escalable desde Laptop a Cliente/Servidor	Si	¿?
Soporte para NetWare y Windows Nt	Si	¿?
Alto Volumen /Bajo Coste de licencias	Si	¿?
Pequeña huella en memoria	Si	¿?
Recuperación automática	Si	¿?

Con uno de los más bajos costes de propiedad disponibles y el rendimiento y potencia requeridos para competir en el mercado actual, Pervasive SQL "es UNICO en su clase".

☐ Deseo más información ☐ Deseo asistir a una presentación
Nombre : _____
Compañía : _____
Dirección : _____
Población : _____
C.Postal y Provincia : _____
Tfno : _____ Fax : _____
Email : _____
Lenguaje programación : _____

Danysoft Internacional
Avda. España 17
28100 Alcobendas - Madrid
Tfno. (91) 6638683
Fax. (91) 6546382
Email : info@danysoft.com
www.danysoft.com

+información en www.danysoft.com

Tabla 4.- Constantes de Inputmode

Constante	Valor	Descripción
ComInputModeText	0	(Predeterminado) Los datos se recuperan como texto mediante la propiedad Input.
ComInputModeBinary	1	Los datos se recuperan como datos binarios mediante la propiedad Input.

Tabla 5.- Mensajes de error

Nº error	Descripción del mensaje
8000	Operación no válida mientras esté abierto el puerto
8001	El valor de tiempo de espera debe ser mayor que cero
8002	Número de puerto no válido
8003	Propiedad disponible sólo en tiempo de ejecución
8004	Propiedad de sólo lectura en tiempo de ejecución
8005	El puerto ya está abierto
8006	El identificador del dispositivo no es válido o no está permitido
8007	La velocidad en *baudios* del dispositivo no está permitida
8008	El tamaño en *bytes* especificado no es válido
8009	Los parámetros predeterminados son incorrectos
8010	El *hardware* no está disponible (está bloqueado por otro dispositivo)
8011	La función no puede asignar las colas
8012	El dispositivo no está abierto
8013	El dispositivo ya está abierto
8014	No se puede activar la notificación de comunicación
8015	No se puede establecer el estado de comunicación
8016	No se puede establecer la máscara de eventos de comunicación
8018	Operación válida sólo cuando el puerto está abierto
8019	Dispositivo ocupado
8020	Error al leer el dispositivo de comunicación

```
Number$ = InputBox$("Escriba el número de
    teléfono:", Number$)
  If Number$ = "" Then Exit Sub
Temp$ = Status

Status = "Marcando - " + Number$
```

Marca el número de teléfono seleccionado.

```
Dial Number$

DialButton.Enabled = True
QuitButton.Enabled = True
CancelButton.Enabled = False

Status = Temp$
End Sub

Private Sub Form_Load()
```

Al establecer "0" en **InputLen,** se indica a *MSComm* que lea todo el contenido del *buffer* de entrada cuando se use la propiedad **Input.**

```
  MSComm1.InputLen = 0
End Sub
Private Sub QuitButton_Click()
```

En este punto, damos por concluida la finalización del programa.

```
  End
End Sub
```

Como complemento al programa, os aconsejo que incluyáis una pequeña base de datos para que la selección de los números esté clasificada y sólo sea necesario un *click* de ratón para introducir el número a marcar.

LA PROPIEDAD COMMEVENT

Devuelve el evento o el error de comunicación más reciente. Esta propiedad no está disponible en tiempo de diseño y es de sólo lectura en tiempo de ejecución.

objeto.CommEvent

Aunque el evento *OnComm* se genera siempre que se produce un error o un evento de comunicaciones, la propiedad *CommEvent* almacena el código numérico de dicho error o evento. Para determinar el error real que ha ocasionado el evento *OnComm*, debemos consultar la propiedad *CommEvent*.

Es aconsejable tener una pequeña base de datos para que los números estén clasificada

La propiedad *CommEvent* devuelve una de las constantes que veremos en las tablas sucesivas.

Constantes del control MSComm

A continuación se detallan todas las constantes de *MSComm* para poder sacarle todo el partido posible a sus características:

Mensajes de error

La siguiente tabla muestra los errores interceptables del control **MSComm.** Con la ayuda de *On Error* y la variable de sistema **Error$,** es posible visualizar un cuadro de diálogo indicándonos la descripción del error (si lo hubo) basándose en la tabla siguiente:

La ventaja de estos programas radica en la velocidad, ya que la llamada se realiza punto a punto

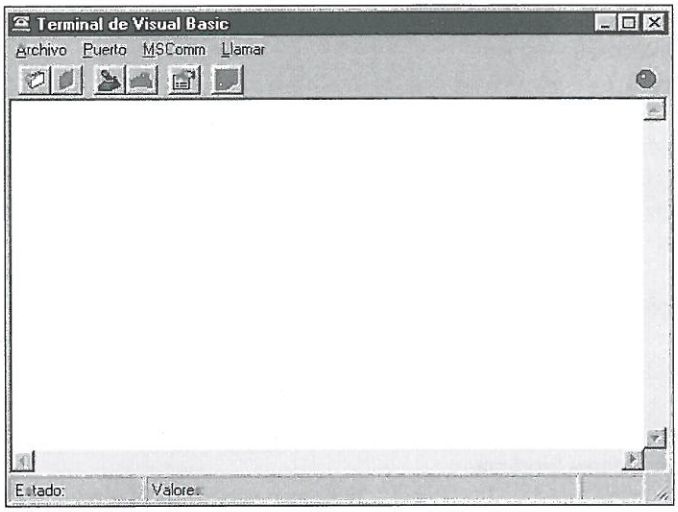
En el siguiente ejemplo, se muestra el código necesario para visualizar uno de los errores anteriores (el código debe estar dentro de un procedimiento):

```
On Error Resume Next
  MSComm1.PortOpen = True
  If Err Then
    MsgBox Error$, 48
    Exit Sub
End If
```

En este ejemplo hemos intentado abrir el puerto de comunicaciones. En caso de que ocurra algún error, se mostrará un mensaje de diálogo con el error retornado mediante la ayuda de la variable **Error$** (No hace falta "dimensionar" **Error$** puesto que está predefinida en el sistema).

CREACIÓN DE UN PROGRAMA TERMINAL

El siguiente proyecto que vamos a abordar en este número será la construcción de un programa terminal. Este tipo de programas es muy parecido a los usados en MS-DOS para las conexiones con BBS's. Con un terminal podemos transferir ficheros, chatear, etc. La diferencia que existe entre un programa terminal y un programa basado en Internet radica en el protocolo utilizado, y sobre todo, en el coste de la llamada.



Vista preliminar del programa terminal.

Un Terminal permite la conexión directa entre dos equipos para transferir archivos y/o mensajes

En los programas de tipo terminal, se busca conectar con un teléfono determinado, y no con un usuario que esté conectado a Internet en un momento dado. Las ventajas de este tipo de programas radica en la velocidad, puesto que la llamada se realiza punto a punto, y no requiere de intermediarios en el envío y recepción de datos.

En la siguiente imagen podéis apreciar el aspecto multifuncional del programa que vamos a crear. El proceso de aprendizaje será un poco largo puesto que se trata de una aplicación para programadores avanzados.

El proyecto consta de los siguientes formularios y módulos:

Formularios

1. **Cansend.frm:** es un cuadro de diálogo que permite al usuario cancelar una operación "Transmitir archivo de texto". Se trata de un formulario no modal que actúa como modal permitiendo continuar a otros procesos.

En el próximo número completaremos el programa con los restantes formularios

2. **Frmprops.frm:** es el formulario encargado de establecer las configuraciones por defecto.

3. **Vbterm.frm:** es el formulario principal que muestra la interfaz del programa.

Módulos

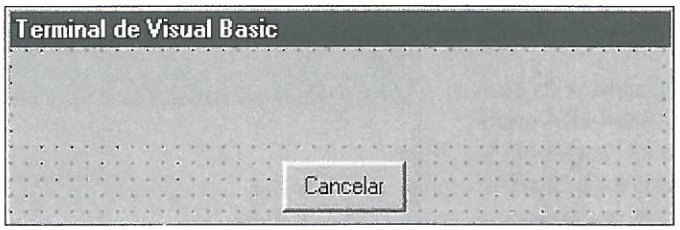
1. **Vbterm.glo:** declara ciertas variables y llama a una DLL del API Win32.

En la siguiente imagen podéis ver un extracto del entorno de Visual Basic que contiene este proyecto:

En el formulario principal, **Vbterm.frm**, se han incluido los siguientes componentes *ActiveX*:

- Imagelist: contendrá todos los gráficos que se incrustarán en los botones de acceso rápido.
- Timer: 2 *Timers* para gestión de eventos.
- Commondialog: para los cuadros de diálogo.
- MSComm: de sobra conocido por vosotros.
- Toolbar: para insertar los botones de acceso rápido a diferentes funcionalidades.
- Menu: para establecer todas las operaciones que se pueden realizar con el programa.
- Textbox: para visualizar datos cuando haya conexión con otro equipo.

Comenzamos el desarrollo del programa con la programación del módulo



Cuadro de diálogo personalizado del terminal.

Vbterm.glo:

Variables públicas. Indicador de echo (Luz) Encendido/Apagado

```
Public Echo As Boolean
```

Indicador para parar el envío de un archivo de texto

```
Public CancelSend As Integer
```

La siguiente instrucción permite utilizar una función del API WIN32 para gestionar la apariencia de un formulario personalizado:

```
Declare Sub SetWindowPos Lib "user32" (ByVal
    hWnd As Long, ByVal hWndInsertAfter As
    Long, ByVal X As Long, ByVal Y As Long,
    ByVal cx As Long, ByVal cy As Long, ByVal
    wFlags As Long)
```

Pasamos al desarrollo de uno de los formularios utilizados en el proyecto, **Cansend.frm:**

```
DefInt A-Z
Option Explicit
```

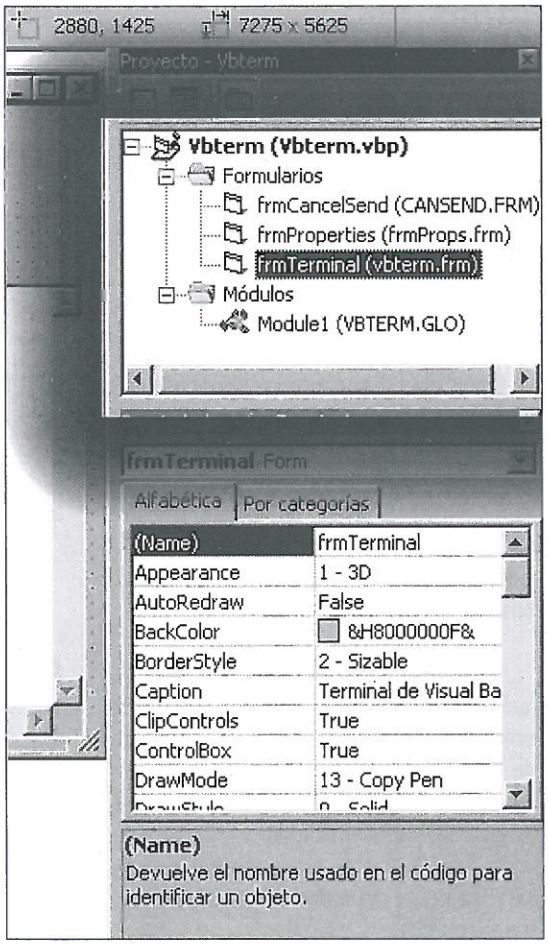
Estas constantes son utilizadas en la función **SetwindowPos** empleada más adelante. Se encargan de definir el comportamiento del formulario al que se aplica:

```
Const SWP_NOMOVE = &H2
Const SWP_NOSIZE = &H1

Private Sub Command1_Click()
  CancelSend = True
End Sub

Private Sub Form_Activate()
```

Hace que el formulario sea una ventana flotante siempre superpuesta.

```
  SetWindowPos hWnd, -1, 0, 0, 0, 0,
      SWP_NOMOVE Or SWP_NOSIZE
End Sub

Private Sub Form_Deactivate()
  If Not CancelSend Then
    frmCancelSend.Show
  End If
End Sub
```

En la imagen se puede observar el formulario utilizado en modo de diseño:



Estas son las preferencias del formulario principal que muestra la interfaz.

En el próximo número completaremos el desarrollo del programa con los restantes formularios que lo componen.

Internet Information Server (II): Seguridad en el servidor con SSL

Enrique Díaz Trobo (Enriqued@jet.es)

Tanto si hemos dado salida a nuestro servidor directamente en Internet como si tenemos solamente una intranet en la que ofrecemos información meramente departamental, será necesario que prestemos una especial atención a las implicaciones de seguridad que esto conlleva.

La razón de esta atención a la seguridad no es sólo para protegernos de accesos malintencionados, sino también procurar que la información que publicamos llegue sólo a quien debe llegar y no a todo el mundo, indiscriminadamente.

Nuestro primer punto de control serán los accesos: establecer quién accede a qué recursos y con qué permisos accederá a ellos. Esto podemos hacerlo con la seguridad propia de Windows NT.

De este modo estableceremos los permisos de usuario sobre las carpetas y recursos del servidor, controlándolo mediante el requisito de las cuentas de usuario y las contraseñas asignadas.

En este sentido, podemos hacerlo también con el IIS, que permite establecer directorios virtuales con permisos de sólo lectura o de ejecución.

Otra manera de denegar accesos mediante el IIS es impedir el acceso de ciertos equipos, esto es, impedir el acceso a ciertas direcciones IP.

CÓMO FUNCIONA IIS

En la mayoría de los casos, necesitaremos permitir accesos anónimos (peticiones que atenderemos pese a que la petición del cliente no presenta nombre de usuario ni contraseña) a nuestro servidor de Internet.

El IIS considera como peticiones anónimas los siguientes casos:

- Un cliente FTP que inicia una sesión y presenta **anonymous** como nombre de usuario.
- Todas las peticiones de Gopher.
- La petición de un explorador de web que no contiene un nombre y una contraseña de usuario en el encabezado HTTP (es la forma más normal del hacerlo).

Cuando recibamos una petición de un explorador, *Internet Information Server* realizará las siguientes comprobaciones antes de atenderla:

- La petición procede de una dirección IP permitida.
- La petición procede de un usuario permitido.
- El usuario tiene permisos para acceder al servidor de Internet.
- El usuario tiene permisos para acceder a las carpetas que contienen la información.

En artículos anteriores dejamos lista la configuración básica del programa Internet Information Server (IIS).
En este artículo veremos algunos aspectos avanzados de su configuración y trataremos el espinoso asunto de la seguridad en los servidores.
Sin duda un tema de total actualidad.

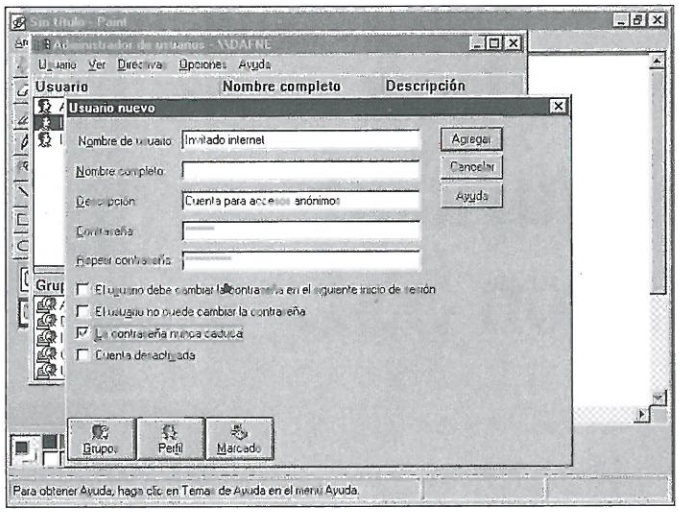


Figura 1. En la imagen se muestra el Administrador de Usuarios para Dominios.

Si cualquiera de las anteriores comprobaciones fallara, la petición sería rechazada.

INICIOS DE SESIÓN ANÓNIMOS

Los inicios de sesión anónimos se realizan a través de una cuenta llamada **IUSR_nombredeequipo**, ya que lo que en realidad se hace es autentificar con esta cuenta a los usuarios que no presentan un nombre de usuario; es decir si se presenta una petición que no tiene nombre de usuario, Windows NT asume que ese usuario es **IUSR_dafne**, pongamos por caso. Por tanto, esta cuenta contendrá los permisos necesarios para acceder al servicio de web, de tal forma que nos permitirá a la vez establecer un límite en las acciones que puede efectuar un usuario durante una sesión anónima.

La creación de la cuenta **IUSR_equipo** tiene las siguientes particularidades:

- Si se instala IIS en un controlador principal de dominio o en un controlador secundario, la cuenta se crea en la base de datos de cuentas de usuario del dominio.
- Si se instala IIS en un servidor miembro del dominio o en un servidor independiente, la cuenta se crea en el equipo local.

Si instalamos IIS en varios controladores del mismo dominio, se crea una cuenta de usuario separada en la base de datos de usuarios del dominio para cada equipo servidor de Internet. Esto no supone ningún problema pero hará más engorrosa la administración, pues tendremos que controlar los accesos y permisos de varias cuentas. Resulta más práctico definir una única cuenta en el dominio que se utilice para todos los controladores del dominio que tengan IIS instalado.

Para ello, crearemos una nueva cuenta mediante el *Administrador de usuarios para dominios*. Haremos a esta cuenta miembro del grupo **Invitados** y le otorgaremos una contraseña, ya que la cuenta de acceso anónimo debe tenerla obligatoriamente. También le otorgaremos el derecho para iniciar sesión en modo local.

En el *Administrador de servicios de Internet* utilizaremos posteriormente el nombre y contraseña de esta cuenta como inicio de sesión anónimo en todos los equipos con IIS instalado. Debemos tener cuidado de que los valores que introduzcamos coincidan exactamente con los valores establecidos en el *Administrador de usuarios para dominios*.

USUARIOS AUTENTIFICADOS

Si no deseamos tener invitados anónimos a nuestro web, podemos restringir su uso a clientes autentificados, esto es, usuarios que proporcionan un nombre de usuario y una contraseña válidos. Si utilizamos la autentificación tendremos la posibilidad de distinguir entre usuarios y mostrar partes específicas de nuestro servidor, ayudados por los permisos NTFS.

Para ello, dispondremos de dos formas de autentificación: **Básica** y **Desafío/Respuesta** de Windows NT. Con la autentificación básica no se codifican las transmisiones entre el cliente y el servidor, que se envían como texto. De este modo, cualquiera que estuviera 'escuchando' en nuestra red, podría conseguir rápidamente un buen número de nombres de usuario con sus respectivas contraseñas.

La clave de la autentificación Desafío/Respuesta cifrará automáticamente tanto el nombre de usuario como su contraseña. Para habilitar los diferentes tipos de autentificación, sólo debemos marcar las casillas correspondientes en el *Administrador de servicios de Internet.*

Otra posibilidad interesante es permitir todos los tipos de conexiones: anónimas, autentificación Básica y Desafío/respuesta. Veamos cómo interactúan estos métodos de conexión:

- Si no permitimos la autentificación con nombre de usuario y contraseña pero sí las conexiones anónimas, una petición de cliente que tenga nombre de usuario y contraseña se tratará como una conexión anónima, despreciando el nombre y contraseña de usuario.
- Si tenemos activada la autentificación y los accesos anónimos, en caso de recibirse una petición que contiene credenciales, no se

utilizarán los valores del Inicio de sesión anónimo y se emplearán el nombre de usuario y contraseña proporcionados con el cliente. Esto se hará, además, por el canal más seguro posible. Esto es, si tenemos activadas la autentificación Básica y Desafío/Respuesta, se utilizará esta última si el navegador lo permite. Si, utilizando credenciales, se intenta acceder a un recurso para el que el usuario no tiene permisos, la petición fallará y se le devolverá una notificación de error al cliente.

- Si una petición anónima falla por no disponer de permisos para acceder al recurso, se enviará al cliente los esquemas de autentificación disponibles con el servicio de web. Si se dispone en ese momento de autentificación básica de HTTP se mostrará en el explorador de web un cuadro de diálogo para suministrar al sistema un nombre de usuario y contraseña, realizando nuevamente la petición, esta vez con credenciales. Esta característica suele utilizarse junto con seguridad SSL (ahora iremos con ella) para asegurar que tanto el nombre de usuario como la contraseña se codificarán antes de su transminisión.

- Si el caso anterior se produce con un explorador que soporte la autentificación Desafío/Respuesta de Windows NT, entonces el explorador enviará automáticamente un nombre de usuario y una contraseña codificadas. La petición se volverá a procesar con los datos suministrados.

AYUDARNOS CON NTFS

Windows NT sin NTFS es como un jardín sin flores. A parte de esta razón (de peso evidente, dicho sea de paso), será conveniente adoptar el formato NTFS, si vamos a tener una intranet y sobre todo si vamos a salir a Internet. NTFS se convertirá en un aliado inestimable para la seguridad de nuestra red.

Queda claro pues que debemos poner los archivos en una partición NTFS. NTFS proporciona seguridad y control de acceso a los archivos de datos. La gracia está en aplicar Listas de control de acceso (ACL) a los archivos de datos para cualquier servicio de publicación de Internet.

Las ACL conceden o deniegan el acceso a la carpeta o al archivo asociado mediante cuentas de usuario o grupos de usuarios específicos de Windows NT.

Cuando un servicio de Internet intenta leer o ejecutar un archivo en nombre de la petición de un cliente, la cuenta de usuario ofrecida por el servicio debe tener permiso, según lo determina la ACL asociada con el archivo.

Si la cuenta de usuario no tiene permiso de acceso al archivo, la petición falla y se devuelve una respuesta, informando al cliente de que se ha denegado el acceso.

Por otra parte, si hay conflictos de algún tipo entre los valores de la partición NTFS y los de Microsoft Internet Information Server, se utilizarán los más estrictos.

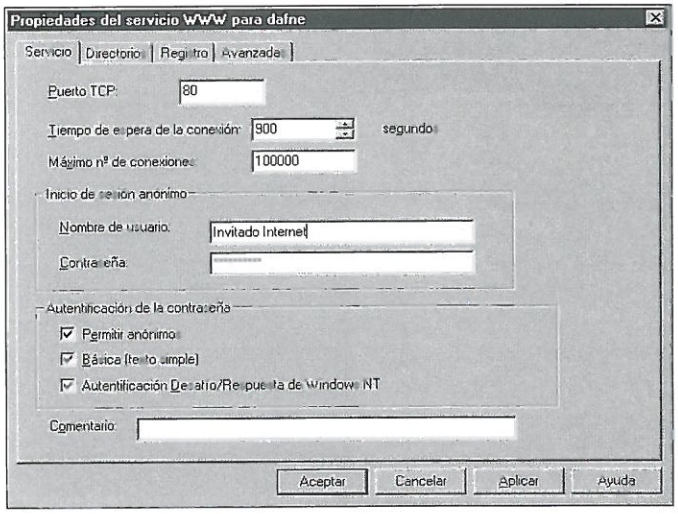


Figura 2
Es necesario habilitar las distintas autentificaciones marcando casillas en el Administrador, del que se ofrece un ejemplo en la imagen.

EL ADMINISTRADOR DE CLAVES

Una vez establecidos los accesos y cómo se llevarán a cabo, es posible que sea necesario establecer seguridad en la transmisión de los datos, bien sea porque son datos sensibles o bien porque nos hemos apuntado a eso que se ha dado en llamar *e-comerce,* comercio electrónico.

IIS ofrece un protocolo para proporcionar seguridad de datos por niveles entre sus protocolos de servicio (el HTTP) y TCP/IP. Este no es otro que el protocolo de seguridad SSL (*Secure Sockets Layer*). SSL proporciona la codificación de los datos, autentificación del servidor e integridad de los mensajes durante una conexión TCP/IP.

Además proporciona un protocolo de intercambio de seguridad que se utiliza para iniciar la conexión TCP/IP. Este intercambio tiene como resultado que el cliente y el servidor se pongan de acuerdo sobre el nivel de seguridad que utilizarán, y llevar a cabo cualquier requisito de autentificación para la conexión.

A partir de entonces, el único papel de SSL es codificar y descodificar el flujo de datos del protocolo de aplicación que se esté utilizando. Es por ello que toda la

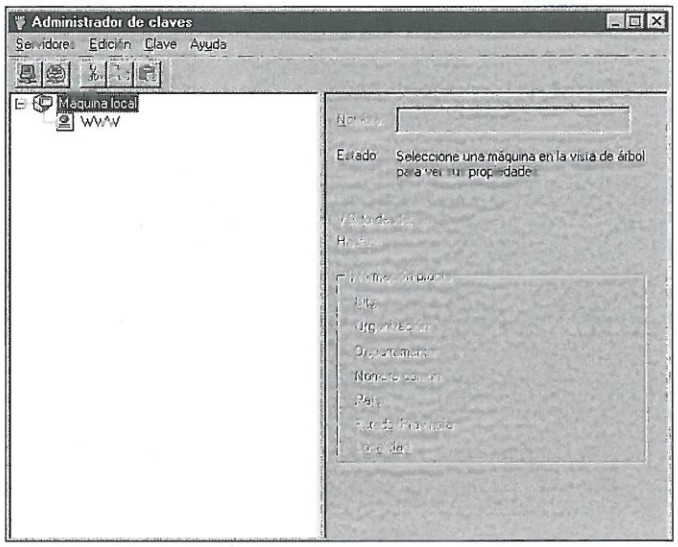


Figura 3
Esta es una de las formas en que se puede activar la seguridad SSL en un servidor.

información, tanto en las peticiones como en las respuestas, queda totalmente codificada, incluyendo la dirección que solicita el cliente, el contenido de los formularios, toda la información relativa a la autorización del acceso, nombres de usuario y contraseñas; así como toda la información que devuelve el servidor al cliente.

Debemos solicitar un certificado antes de poder utilizar SSL

Si habilitamos SSL en el servidor podemos enviar y recibir informaciones con total confidencialidad a través de Internet. Esta seguridad tiene su precio en la rapidez, ya que las conexiones SSL son más lentas. Así pues, lo más recomendable es utilizarlo sólo donde sea realmente necesario, como son las autorizaciones de pagos o carpetas con información delicada.

ACTIVAR SSL

Para activar la seguridad SSL en un servidor que vaya a salir a Internet debemos generar un archivo de claves y un archivo de petición. Una vez hecho esto debemos solicitar un certificado a una Autoridad de certificación, tras lo cual lo instalaremos en el servidor y activaremos la seguridad SSL en una carpeta de servicio WWW.

Para ello ejecutaremos la utilidad *Administrador de claves*. También podemos hacerlo desde el *Administrador de servicios de Internet*, ejecutando la opción *Administrador de claves* bajo el menú **Herramientas**. Veremos una pantalla como la que se muestra en la figura 3. Abriremos el menú **Clave** y seleccionaremos **Crear nueva clave**. En la pantalla de la figura 4 rellenaremos las siguientes opciones:

- *Nombre de la clave*. Es el nombre específico que asignaremos a la nueva clave que estamos generando en este momento.
- *Contraseña*. Contraseña que se utilizará para codificar la clave privada.
- *Bits*. Por defecto está a 512, con lo que se generará un par de claves con una longitud de 512 bits.
- *Organización*. Indicamos el nombre de la organización o compañía. El cliente podrá verificar su conexión mediante este campo.
- *Departamento*. Nombre del departamento, si lo hay.
- *Nombre común*. Daremos el nombre del dominio del servidor, por ejemplo **diazmarcos.org**.
- *País*. Denominación ISO de nuestro país. En nuestro caso: **es**.
- *Estado*. Estado o provincia donde se encuentra la organización.
- *Localidad*. Ciudad de la compañía.
- *Archivo de*. Indicamos el archivo de petición que se creará.

Una vez hayamos completado la información, pulsaremos **Aceptar**, debiendo introducir la contraseña que escribimos anteriormente.

Tras hacer esto se creará la clave y un icono en la pantalla principal (figura 5). La clave que se ha generado no será válida para su utilización en Internet, por lo que tendremos que obtener un certificado de una Autoridad de certificación. Para obtener la certificación podemos dirigirnos a **http://www.verisign.com** o bien a

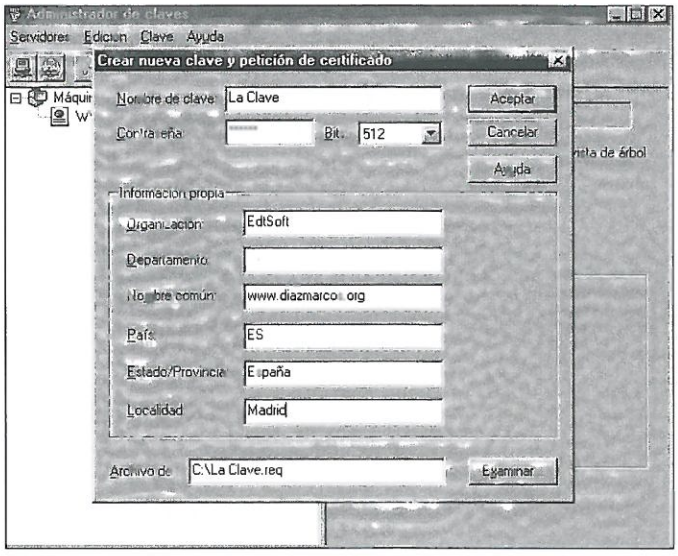


Figura 4
Indica el archivo de petición que se creará.

Banesto, en **http://www.banesto.es**, que es Autoridad de Certificación Española

Después de haber realizado la petición del certificado, recibiremos un certificado firmado de la Autoridad de certificación. Para instalarlo ejecutaremos el *Administrador de claves* y seleccionaremos el icono de claves correspondiente a ese certificado. En el menú **Clave** seleccionamos y abrimos el archivo de certificado que nos enviaron. De nuevo será necesario introducir la clave y una dirección IP donde instalar el certificado. Si no especificamos ninguna, el certificado se aplicará a todos los servidores virtuales creados en el sistema.

El precio de la seguridad y confidencialidad está en la pérdida de rapidez en los sistemas

Ya sólo nos queda hacer que se solicite una clave SSL. Para ello accederemos a las propiedades del servicio WWW en el *Administrador de Servicios de Internet* y activaremos la opción **Requiere asegurar canal SSL** en aquellas carpetas que requieran seguridad SSL.

LAS CERTIFICACIONES

¿Por qué las certificaciones? Imaginemos que un "avispadillo" se baja o imita las páginas de El Corte Inglés, las pone en su servidor, instala SSL, y se dedica a esperar a que la clientela le deje amablemente los números de sus tarjetas de crédito. La seguridad SSL no valdría de nada.

Es por esta razón por la que es necesaria una Autoridad de Certificación que se ocupe de autentificar a los participantes en transacciones o comunicaciones dentro de Internet. La Autoridad de certificación tiene la responsabilidad de emitir certificados a los usuarios, ya sean éstos clientes o servidores. Es de este modo como se garantiza que las transacciones sean realmente seguras. Se trata de que un tercer organismo de fe de la identidad de los participantes.

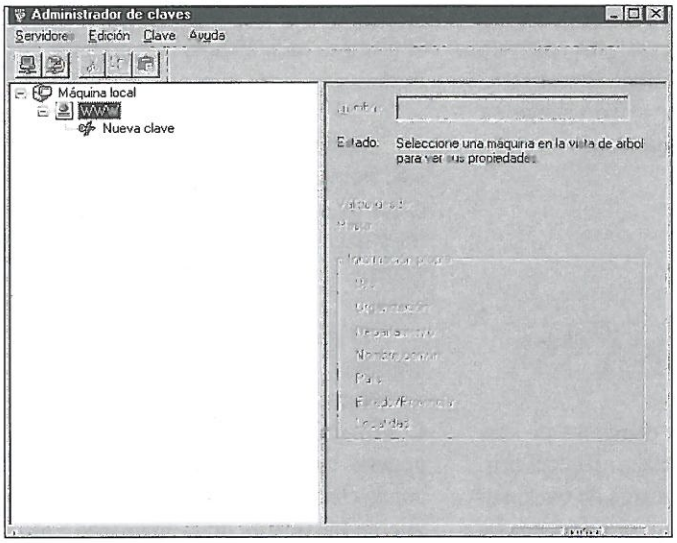

Administrador de claves
Servidores Edición Clave Ayuda
Máquina local
WWW
Nueva clave
Estado: Seleccione una máquina en la vista de árbol para ver sus propiedades

Figura 5
En esta imagen se puede observar la pantalla principal, en la que se generará una clave para su utilización en Internet.

SSL proporciona autentificación, seguridad e integridad en las transaciones

De este modo establecemos la *autentificación*, esto es, la garantía de que el servidor web al que llamamos sea quien realmente dice ser.

El protocolo SSL se encargará de asegurar:

- La *integridad*. La garantía de que los mensajes que enviamos o recibimos no han sido modificados.
- La *confidencialidad*. Nadie sin autorización puede leer la información transmitida; y garantiza que efectivamente la recibe quien deba recibirla.

IDENTIFICAR UN SERVIDOR SEGURO

Una de las formas más evidentes es ver la información relativa a los certificados. Netscape y Microsoft Explorer, tienen opciones para mostrar la información relativa al nivel de seguridad del servidor, su certificación y la autoridad de certificación que la emitió. Aquí podremos examinar el certificado del servidor. Debemos comprobar que el nombre del ordenador y el nombre de la compañía coinciden con los nombres con los cuales hemos pretendidamente conectado.

En el caso del ejemplo anterior, si viéramos algo extraño o errores supuestamente tipográficos (por ejemplo que pusiera "El Corte Ingels" en lugar de "El Corte Inglés"), debemos sospechar siempre e intentar verificar la identidad por otros medios, como una llamada telefónica, o un correo electrónico a una dirección que sepamos que es buena.

Otro modo de saber que hemos entrado un servidor seguro es que cambia el identificador de URL de http a https (*Hipertext Transport Protocol Secure*). Además, en el navegador veremos algún icono identificativo de este hecho, una llave en Netscacape y un candado en

Microsoft Explorer (como podéis ver, andan como el perro y el gato hasta en esto). Por otra parte, ambos navegadores muestran mensajes de aviso cuando se entra en un lugar seguro.

CÓMO FUNCIONA SSL

El protocolo SSL se compone de dos capas: una, la de más bajo nivel (el *SSL Record Protocol)*, se encarga de encapsular los protocolos de nivel más alto. La segunda, *SSL Handshake Protocol*, se encarga de la negociación de los algoritmos de encriptación, así como la autentificación de cliente y servidor.

Durante la conexión inicial el cliente envía el mensaje **Client-hello**, y adjunta información sobre los sistemas criptográficos de los que dispone, siendo el primero en la lista el que prefiere utilizar.

El servidor responderá con el mensaje **Server-hello**, y adjunta un identificador de conexión, su clave certificada e información sobre los sistemas criptográficos que soporta.

El cliente elegirá un sistema criptográfico, verificará la clave pública del servidor para responder con el mensaje **Client-master-key**. Aquí es donde se genera una clave aleatoria encriptada con la clave pública del servidor. Este es uno de los pilares de la seguridad SSL.

Aún en el caso de que alguien consiguiera desencriptar la información (cosa muy difícil) sólo sería útil para 'reventar' esa conexión. Una ulterior conexión entre el mismo cliente y el mismo servidor tendría una clave criptográfica diferente.

Tras la generación de la clave el cliente enviará el mensaje **Client-finished**, en el que se envía un identificador de conexión ya encriptado con su clave. El servidor responde con un mensaje **Server-verify**, que autentifica su identidad junto con una clave de escritura generada a partir del mensaje **Client-master-key**.

El servidor puede realizar una última verificación enviando al cliente un mensaje **Recuest Certificate**. Si todo va bien el cliente responderá con un mensaje **Client-Certificate**, esto es, su certificado.

Ya está todo hecho y finalizada la práctica. El servidor envía un mensaje **Server-Finish** y se cede el control a los protocolos de nivel aplicación. SSL permanece por debajo encriptando todo el flujo de información de las transacciones con las claves que se han generado durante la negociación.

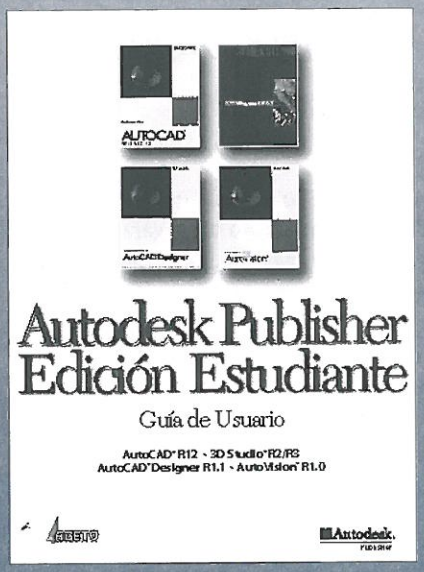
AUTODESK PUBLISHER EDICIÓN ESTUDIANTE

¿QUÉ PROGRAMAS INCLUYE?

- AutoCAD R12
- 3D Studio R2/R3
- AutoCAD Designer R1.1
- Autovision R1.0.

Y además:

Más de 700 páginas de lecciones de Autodesk con guías de referencia y documentación software on-line



Formato 21x29,7

TODO LO QUE NECESITA PARA EL DISEÑO Y VISUALIZACIÓN PROFESIONAL EN 2D Y 3D LO HALLARÁ EN ESTE PAQUETE DE PROGRAMAS.

EDICIÓN ESPECIAL PARA ESTUDIANTES
19.500 PTAS. (I.V.A. INCLUIDO)

DE VENTA EN LIBRERÍAS Y DISTRIBUIDORES AUTODESK

Programación de DirectX con Delphi (V)

Constantino Sánchez Ballesteros (constantino@redestb.es)

Continuando con temas anteriores, en este artículo se comenzará con los detalles internos de los ficheros que han sido creados con la librería de imágenes. Se definirá el FX Editor, describiendo por otro lado la programación de arcades de carreras

Con el fin de que los lectores puedan tener control absoluto sobre los datos almacenados en los diferentes tipos de ficheros utilizados, lo primero que se va a ofrecer en este artículo son todos los entresijos de los archivos resultantes de la librería de imágenes, comentada en números anteriores:

Header del archivo

Identifer	array[0..3] of char = "IL01"
ImageCount	Integer
TransparentColor	Integer
Palette	TImgLibPalette

El campo *Identifer* indica que el fichero se corresponde con uno compatible de la librería de imágenes. *ImageCount* contiene el número de imágenes asignadas a dicha librería. *TransparentColor* representa el color que será tomado como transparente al efectuar el volcado de gráficos al *backbuffer*. Por último, *Palette* contiene los colores utilizados en las imágenes (del 0 al 255).

La siguiente cabecera (*header)* incluye los datos previos a los gráficos (ancho, alto,...), de tal forma que cada *Bitmap* viene precedido por un *header*:

Header del bitmap

Width	Integer	(Ancho del bitmap en pixels)
Height	Integer	(Alto del bitmap en pixels)
Name	String[8]	(Nombre de la imagen)

Después de este *header*, aparecen los datos gráficos que realmente se visualizarán en la pantalla. El número de *bytes* que siguen la cabecera del *bitmap* se calculan del siguiente modo:

```
WidthBytes * Height
```

Los *bits* de la imagen son palabras alineadas, y en consecuencia, el *width* en *pixels* debería ser menor que el *width* en *bytes*. Para calcular el *width* en *bytes* utilizaremos la función **WidthBytes** de la unidad *BmpUtil* .

Un ejemplo de lo comentado puede ser:

```
BmpSizeInBytes := WidthBytes(PixelWidth) *
    PixelHeight.
```

EL "FX EDITOR" O LIBRERÍA DE INTROS

Una de las posibilidades más sorprendentes de la librería DGC es su editor de efectos para transiciones de pantallas. Las transiciones suelen utilizarse al comienzo del juego para presentar menús de ayuda, créditos, información adicional, cambio entre fases del juego, (y todo lo que se os pueda ocurrir al respecto...).

Tabla 1.- Tipos de transiciones

No transition effect:	No efectúa ningún tipo de transición.
Slideright:	Efectúa un deslizamiento del *bitmap* hacia la derecha.
Slideleft:	Efectúa un deslizamiento del *bitmap* hacia la izquierda.
Slideup:	Efectúa un deslizamiento del *bitmap* hacia arriba.
Slidedown:	Efectúa un deslizamiento del *bitmap* hacia abajo.
Zoomopen:	La apertura del gráfico se realiza mediante un efecto zoom.
Curtains:	Efecto de cortinilla.
Shutters:	Efecto de ventanilla.
Quarters:	Aparición del *bitmap* como si fuera la aguja del reloj.
Verticallines:	Fundido con líneas verticales.
Horizontallines:	Fundido con líneas horizontales.
Updownlines:	Fundido con líneas de arriba a abajo.
Leftrightlines:	Fundido de izquierda a derecha.
Scrollright:	Scrolling del gráfico hacia la derecha.
Scrollleft:	Scrolling del gráfico hacia la izquierda.
Scrollup:	Scrolling hacia arriba.
Scrolldown:	Scrolling del gráfico hacia abajo.
Drip:	La transición se efectúa derritiendo un gráfico mientras aparece el otro (realmente espectacular).
Horizontalopenout:	Apertura del gráfico hacia fuera de forma horizontal.
Verticalopenout:	Apertura del gráfico hacia fuera de forma vertical.
Diagonaltopleft:	Efectúa una diagonal comenzando desde la parte superior hasta la diagonal inferior izquierda de la pantalla.
Diagonaltopright:	Efectúa una diagonal comenzando desde la parte superior hasta la diagonal inferior derecha de la pantalla.
Diagonalbottomleft:	Efectúa una diagonal comenzando desde la parte inferior hasta la diagonal superior izquierda de la pantalla.
Diagonalbottonright:	Efectúa una diagonal comenzando desde la parte inferior hasta la diagonal superior derecha de la pantalla.
Verticalstripes:	El gráfico aparece mediante múltiples líneas verticales.
Horizontalstripes:	El gráfico aparece mediante múltiples líneas horizontales.
Largeverticalstripes:	Visualización a través de cuatro líneas verticales que muestran el gráfico al avanzar de un lado a otro.
Largehorizontalstripes:	Visualización a través de cuatro líneas horizontales que muestran el gráfico al avanzar de un lado a otro.

El funcionamiento básico del programa consiste en cargar una serie de *bitmaps*, tal y como hacemos cuando creamos las librerías de imágenes. Una vez tenemos todos los gráficos que compondrán la serie de transiciones, sólo tenemos que asignar el efecto deseado a cada una de ellas y el tiempo de duración.

A continuación se muestran los tipos de transiciones que podemos crear con el programa:

A continuación veremos los pasos que hay que seguir para crear una librería que contenga los efectos especiales de las pantallas gráficas utilizadas:

1. En primer lugar, abrimos el programa **Intro Library Editor** que se encuentra en el menú **Programas** del botón **Inicio** de Windows.

2. En el menú **File**, seleccionamos **New Library**.

3. Cargamos las pantallas gráficas (*bitmaps*) que necesitemos para realizar las transiciones entre unas y otras.

4. El siguiente paso consiste en definir los parámetros de selección del tiempo y velocidad que durará cada efecto especial aplicado a un *bitmap*. Para ello pincharemos con el ratón en el primero de los gráficos que componen la transición. A continuación, en el cuadro combinado del programa escogeremos el efecto de transición que se aplicará (uno de los comentados en la tabla anterior para cada gráfico). Seguidos del *Combo box*, aparecerán otros dos cuadros de selección; el primero se encarga de definir la velocidad a la que será representada la transición. El segundo cuadro, define el tiempo que tardará el gráfico en pasar al siguiente *bitmap* de los seleccionados anteriormente. En nuestro caso únicamente definiremos los valores numéricos adecuados para nuestros propósitos.

5. Para finalizar nuestro trabajo con el editor, tan sólo queda guardar todo el proceso en un archivo que tendrá la extensión CRL. La mayoría de botones utilizados en el FX Editor tienen la misma utilidad que el editor de banco de imágenes comentado en anteriores números, por lo tanto, no entraré en detalles.

A continuación veremos un uso práctico del manejo de una librería creada en este editor de efectos especiales disponible en DGC:

```
uses
  Windows, Messages, SysUtils, Classes, Graphics,
       Controls, Forms, Dialogs, Dgcfxlib,
       BMPUtil, DGC;
```

En la cláusula **uses** del programa, ha sido necesario incluir la unidad **BMPUtil** para la correcta gestión de los gráficos asignados en el *FX Editor*.

Para el buen funcionamiento del programa, debemos insertar dos componentes visuales en el formulario; el primero es el archiconocido *DGCScreen*. El otro es el relacionado con la librería de FX (*DGCIntrolib*):

```
type
 TForm1 = class(TForm)
  DGCScreen1: TDGCScreen;
  DGCIntroLib1: TDGCIntroLib;
  procedure DGCScreen1Initialize(Sender:
      TObject);
  procedure FormKeyPress(Sender: TObject; var
      Key: Char);
 private
  { Private declarations }
 public
  { Public declarations }
  Procedure DrawFrame;
 end;

var
 Form1: TForm1;

implementation

{$R *.DFM}

procedure TForm1.DGCScreen1Initialize(Sender:
      TObject);

begin
```

Asignamos el nombre de la librería y su localización en el *Path*. En este caso, la librería se llama **Intdem.crl**, y su *Path* es tomado de la dirección del ejecutable:

```
DGCIntroLib1.Filename:=Extractfilepath(Applicati
      on.ExeName)+'\intdem.crl';
```

Reproducimos los datos almacenados en la librería mediante el método *Play*:

```
  DGCIntroLib1.Play;
```

Con el procedimiento *Drawframe* dibujamos como *background* un texto determinado mediante *Textout*. Además, utilizaremos la potencia del componente *DGCIntrolib* para que dicho fondo aparezca con un efecto de transición. Ésta es la forma de crear los efectos manualmente, es decir, sin necesidad de utilizar el *FX Editor*.

Mediante *Appearsurface* conseguimos establecer los parámetros necesarios para la transición: buffer gráfico, tipo de transición, velocidad de representación y tiempo de espera.

```
   DrawFrame;
DGCIntroLib1.AppearSurface(DGCScreen1.Back,z
      oomopen,5,0);
end;

Procedure TForm1.DrawFrame;
begin
   with DGCScreen1.Back.Canvas do
   begin
      Brush.Style:=BsClear;
      Font.Size := 14;
      Font.Color := rgb(255,255,255);
 TextOut(((DGCScreen1.Back.Width div 2) -
      (TextWidth('Presione Escape para volver a
      Windows') div 2)) ,240 , 'Presione Escape
      para volver a Windows');
      Release;
   end;
end;

procedure TForm1.FormKeyPress(Sender:
      TObject; var Key: Char);
begin
   if Key = #27 then
      close;
end;
end.
```

La clase TDGCrace

ARCADES DE CARRERAS

Llega el momento de aprender otro nuevo tipo de DGC muy funcional, la clase TDGCrace. Con ella, podemos crear juegos orientados a arcades de carreras. El ejemplo que expondré a continuación, no es más que un coche sobre un circuito circular que soporta colisiones con los bordes de la carretera. Debido a su simplicidad, es un ejemplo muy ilustrativo para aprender las funciones más importantes de esta clase incorporada en DGC.

El control de colisiones entre el coche y la carretera está muy logrado, puesto que las colisiones se efectúan independientemente del perfil de los *sprites*; es decir, la detección de colisión no tiene por qué hacerse sobre una superficie rectangular. En artículos anteriores hemos efectuado colisiones sobre superficies rectangulares, aunque este sistema no es muy ortodoxo debido a lo irreal que puede resultar cuando un *sprite* no cubre el rectángulo imaginario que lo define.

Con la clase TDGCRace la creación de arcades de coches en 2D es puro trámite

En este ejemplo, la colisión se efectuará cuando un *pixel* se superponga sobre otro *pixel* de un *sprite* o gráfico determinado. Para ello utilizaremos el método *collisiontest*.

Es importante indicar que para que la animación del coche sea lo más suave posible, tendremos que utilizar más *frames* en la animación de los que yo he usado, puesto que de lo contrario, el coche parecerá que rota a trozos y de una forma muy brusca.

Como en cualquier programa de Delphi, lo primero de todo es cargar las unidades correspondientes para la creación del proyecto, además de los controles necesarios. En nuestro caso, como controles del formulario aparecen los del tipo *TDGCScreen* (para la inicialización del modo de vídeo), *TDGCImagelib* (para contener las imágenes que servirán como *sprites*) y *TDGCSpritemgr* (para todo lo relacionado con la gestión de *sprites* en pantalla).

```
uses
 Windows, Messages, SysUtils, Classes, Graphics,
      Controls, Forms, Dialogs,
 DGCspts, DGCILib, DGC;

type
 TForm1 = class(TForm)
  DGCScreen1: TDGCScreen;
  DGCImageLib1: TDGCImageLib;
  DGCSpriteMgr1: TDGCSpriteMgr;
```

```
  procedure DGCScreen1Initialize(Sender:
      TObject);
  procedure DGCScreen1Flip(Sender: TObject);
  procedure FormKeyPress(Sender: TObject; var
      Key: Char);
private
  { Private declarations }
public
  { Public declarations }
end;

var
 Form1: TForm1;

implementation

{$R *.DFM}
```

Una vez seguidos los pasos preliminares, comenzamos con la creación de las animaciones que utilizaremos en el programa mediante el siguiente procedimiento:

```
procedure TForm1.DGCScreen1Initialize(Sender:
      TObject);
begin
 DGCSpriteMgr1.Animations.Add('car', True);
   with DGCSpriteMgr1.Animations[0] do
    begin
      SetFrames(0, 100, [0]);
      SetFrames(2, 100, [1]);
      SetFrames(4, 100, [2]);
      SetFrames(6, 100, [3]);
      SetFrames(8, 100, [4]);
      SetFrames(10, 100, [5]);
      SetFrames(12, 100, [6]);
      SetFrames(14, 100, [7]);
      SetFrames(16, 100, [8]);
      SetFrames(18, 100, [9]);
      SetFrames(20, 100, [10]);
      SetFrames(22, 100, [11]);
      SetFrames(24, 100, [12]);
      SetFrames(26, 100, [13]);
      SetFrames(28, 100, [14]);
      SetFrames(30, 100, [15]);
   end;
```

Mediante la clase *TDGCSpriteMgr* y sus métodos, creamos la animación correspondiente al coche. Para ello, hemos usado *setframes* con los parámetros necesarios. Puesto que en nuestro circuito vamos a tener conos de señalización, debemos crear también sus animaciones:

```
  DGCSpriteMgr1.Animations.Add('Cone', True);
  with DGCSpriteMgr1.Animations[1] do
  begin
    SetAllDirections(100, [17]);
  end;
```

Cuando un cono sea golpeado por nuestro coche, deberá animarse de una forma determinada (rotación sobre su eje). Para ello, debemos crear también esa animación:

```
  DGCSpriteMgr1.Animations.Add('SpinCone',
      True);
  with DGCSpriteMgr1.Animations[2] do
  begin
    SetAllDirections(60, [17,18,19,20]);
  end;
```

En este punto, agregamos el *sprite* correspondiente al coche mediante el método *Addracecar*:

```
  DGCSpriteMgr1.Sprites.AddRaceCar(0, 320,
      360, 8, 4, 0);
  with DGCSpriteMgr1.Sprites[0] as
      TDGCRaceCar do
  begin
    AllActions := laStopInside;
    Handling := 50;
  end;
```

A continuación, asignamos algunos conos sobre la pista del circuito:

```
  DGCSpriteMgr1.Sprites.AddBouncer(1, 70,
      320, 0, 0, 1);
  DGCSpriteMgr1.Sprites.AddBouncer(1, 320,
      70, 0, 0, 1);
  DGCSpriteMgr1.Sprites.AddBouncer(1, 80,
      100, 0, 0, 1);
  DGCSpriteMgr1.Sprites.AddBouncer(1, 280,
      120, 0, 0, 1);
  DGCSpriteMgr1.Sprites.AddBouncer(1, 550,
      160, 0, 0, 1);
  DGCSpriteMgr1.Sprites.AddBouncer(1, 520,
      290, 0, 0, 1);
end;
```

El procedimiento *DGCScreen1Flip* ya es conocido por vosotros; se encarga de transferir los gráficos desde la memoria a la pantalla:

```
procedure TForm1.DGCScreen1Flip(Sender:
      TObject);
var
  sx, sy, n: Integer;
begin
```

Dibujamos el *background*, el que corresponde al circuito.

```
  DGCScreen1.Back.Draw(0, 0,
    DGCScreen1.Images[16], False);
```

Actualizamos los *sprites* mediante el método *update* pero guardando previamente los valores de las coordenadas del coche por si hay una posible colisión con los conos. Estas coordenadas temporales se almacenan en las variables **sx** y **sy**.

```
  with DGCSpriteMgr1.Sprites[0] do
  begin
    sx := x;
    sy := y;

    DGCSpriteMgr1.Update;
    if DGCScreen1.Back.CollisionTest(0, 0,
    DGCScreen1.Images[Frame],
      X, Y, True) then
    begin
      X := sx;
      Y := sy;
      Speed := 0;
    end;
  end;
```

Chequeamos la colisión del coche con los conos. Para los que no conozcáis la función **Pred**, cabe señalar que se utiliza para averiguar el valor precedente al devuelto en la función evaluada. Para más información, consultar el manual de Delphi .

```
  for n := 1 to
    Pred(DGCSpriteMgr1.Sprites.Count) do
  begin
    if DGCSpriteMgr1.Sprites.Collision(n, 0)
    then
    begin
      with DGCSpriteMgr1.Sprites[n] do
      begin
```

En este punto, controlamos que el cono siga la dirección del coche pero a mayor velocidad, cuando se establezca colisión entre ambos:

```
            MaxSpeed := 5;
            Speed := 5;
            Direction :=
      DGCSpriteMgr1.Sprites[0].Direction;
            Animation := 2;
            AllActions := laStopOutSide;
          end;
          break;
        end;
      end;

    //Dibujamos los sprites
    DGCSpriteMgr1.Draw;
end;
```

Con *Formkeypress* acabamos la ejecución del programa al pulsar la tecla correspondiente (en este caso, la tecla **Escape**):

```
procedure TForm1.FormKeyPress(Sender:
      TObject; var Key: Char);
begin
  if Key = #27 then Close;
end;

end.
```

EL PROYECTO FAHRENHEIT

El futuro de las tecnologías referentes al desarrollo gráfico de alto rendimiento, pasa por el proyecto Fahrenheit. Silicon Graphics y Microsoft, han anunciado una estrategia para incrementar las capacidades gráficas de nuestros queridos PC's. Por supuesto, sólo estará disponible, en principio, para Windows y UNIX. Esta iniciativa significará un avance en la tecnología gráfica y creará una arquitectura común que permitirá avanzados y poderosos gráficos en el mercado de los ordenadores compatibles. Debido a que Microsoft decidió "olvidar" OpenGL en favor de DirectX, Silicon Graphics ha requerido que la empresa del señor Gates optimice la versión actual de OpenGL sobre Windows para que el proyecto Fahrenheit siga adelante.

Lo mejor de esta tecnología que está por llegar, es la total compatibilidad con los programas ya creados bajo DirectX. Por lo tanto, no debemos preocuparnos al respecto.

El proyecto Fahrenheit, creará una Suite de API's para la arquitectura múltiple de DirectX sobre el sistema operativo Windows y UNIX. Tal y como he comentado en anteriores ocasiones, un API es una interfaz común con la que los desarrolladores pueden aprovechar todas las capacidades ofrecidas por un ordenador de forma sencilla. Fahrenheit incorporará Direct3D y DirectDraw en conjunción con las tecnologías complementarias de Silicon Graphics (OpenGL, OpenGL Scene Graph y OpenGL Optimizer). Esta arquitectura establecerá las bases para la innovación de gráficos y aplicaciones de visualización incluyendo Internet, juegos, negocios, creación de contenido digital, aplicaciones científicas y médicas, etc.

Fahrenheit representa a medio plazo el futuro de la tecnología gráfica basada en la plataforma PC

Fahrenheit mostrará el camino de un *software* gráfico totalmente escalable que satisfará todas las necesidades de las API´s de bajo nivel para construir escenas gráficas con grandes modelos de visualización y funciones de simplificación en la representación de las mismas.

Esto es algo que la industria ha buscado y necesitado desde hace mucho tiempo. A través de este acuerdo, Silicon Graphics podrá seguir trabajando en la creación de espectaculares API´s gráficos para sistemas basados en Windows.

FAHRENHEIT: DETALLES DE SU ARQUITECTURA

El proyecto Fahrenheit producirá tres componentes en diferentes etapas:

- **Fahrenheit low-level API:** Será el API gráfico primario para las aplicaciones orientadas a consumidores y profesionales sobre el entorno Windows. Este API de bajo nivel, evolucionará desde Direct3D, DirectDraw y OpenGL permitiendo total compatibilidad con aplicaciones y *Drivers* de *hardware* escritos en Direct3D. Además, también permitirá total compatibilidad con la tecnología gráfica de Silicon, OpenGL.

- **Fahrenheit Scene Graph API:** Será el API de alto nivel de Fahrenheit que permitirá aplicaciones basadas sobre los sistemas operativos Windows e IRIX de Silicon Graphics. Este API contendrá el actual Scene Graph para permitir compatibilidad. Scene Graph implementará nuevas estructuras de datos y algoritmos a un alto nivel para incrementar el rendimiento gráfico y asistir al desarrollo de sofisticadas aplicaciones con gran riqueza visual.

- **Fahrenheit Large Model Visualization Extensions:** Se basará en el API gráfico de OpenGL "Optimizer" y el "DirectModel" de Hewlett-Packard y Microsoft. Operará en conjunción con el API Scene Graph anterirmente comentado. Las extensiones del *Large Model Visualization* suman funcionalidades que permitirán la manipulación interactiva de grandes modelos 3D, como automóviles, con todos sus detalles (sí, he dicho bien, "CON TODOS SUS DETALLES"). Este API incorporará funcionalidades como simplificación de resoluciones de objetos múltiples en el API Scene Graph. De este modo, los desarrolla-

Rellena este cupón

CORRE. CADA 3 SEGUNDOS MUERE UN NIÑO EN EL TERCER MUNDO.

☐ SI, DESEO RECIBIR MAS INFORMACION SIN COMPROMISO.

Nombre .....

Dirección .....

Localidad ..... Provincia .....

C.P. ..... Tel. .....

C/ Tutor, 27. 28008 Madrid. C/ Balmes, 32, 3º. 08007 Barcelona.

718 **Tel.: 902 402 404** Ayuda en Acción

Practica el mejor ejercicio para tu corazón: la solidaridad. En Ayuda en Acción, necesitamos que la ejercites para poder impulsar nuestros Proyectos de Desarrollo Integral en el Tercer Mundo. Corre. Rellena este cupón y contribuirás a llevar la esperanza a quien más lo necesita.

Apadrina un niño. Tel.: 902 402 404

Desde 1981
trabajando con el tercer mundo

Ayuda en Acción

dores pueden escribir fácilmente aplicaciones que interactuen con bases de datos de objetos extremadamente grandes.

Recientemente, las compañías involucradas en el proyecto Fahrenheit, han acordado colaborar en el desarrollo de un nuevo Kit (DDK) para la creación de *drivers* que soporten OpenGL sobre Windows 9X y NT. Esperemos que todos estos acuerdos y desarrollos conjuntos se hagan profesionalmente y a corto plazo, puesto que la tecnología Fahrenheit beneficiará tanto a programadores como a usuarios.

DIRECTMUSIC

Microsoft no para de crear tecnología, sobre todo en el apartado multimedia. La empresa creadora de DirectX acaba de lanzar DirectMusic. Con este módulo, podremos crear música de altísima calidad. DirectMusic que permite construir inteligencia musical en los títulos multimedia para Windows. Los diseñadores apreciarán los beneficios del soporte para MIDI y podrán personalizar sus archivos de instrumentos.

La arquitectura es extensible para permitir la mejor integración de cualquier tecnología musical basada en Windows. Hasta hace poco, la única opción rentable para la reproducción musical necesitaba utilizar instrumentos *hardware* para efectuar la composición. Una nueva norma de la industria, llamada DLS o *DownLoadable Sounds*, permite crear una paleta musical personalizada y lograr resultados consistentes sobre el hardware además de sintetizadores por software de cualquier fabricante.

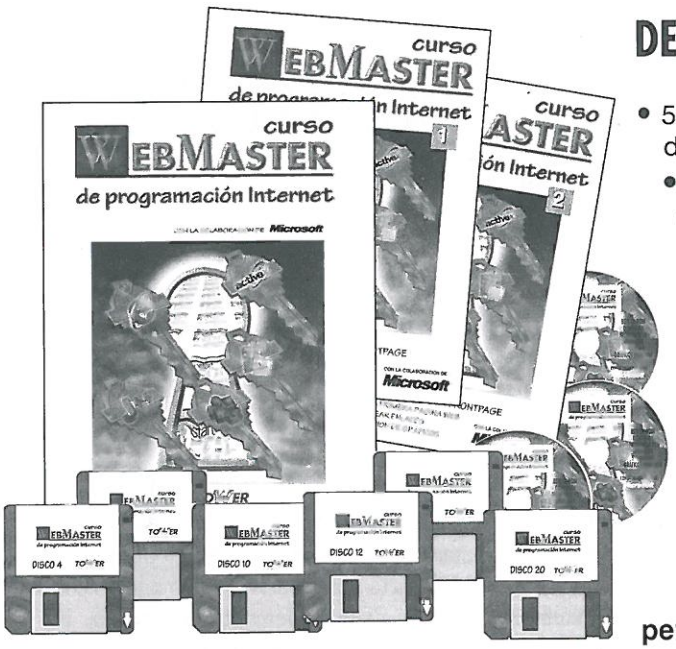
DirectMusic tiene varios métodos para apoyar la norma DLS. Para las plataformas que no posean el sintetizador por hardware, obtendrán uno por software manteniendo un apoyo completo al nivel primario de DLS. Dada la naturaleza flexible de DLS, es posible para una aplicación incluir capacidades avanzadas de los instrumentos, además de efectos de sonido personalizados. Mediante el hardware apropiado, DirectMusic permitirá obtener música de altísima calidad. Como detalles de esta tecnología.

Cabe decir que DirectMusic puede alterar las notas, acordes, balances, texturas, tempos, etc. en tiempo real.

DLS permite especificar timbres exactos en las colecciones de sonidos creadas. Podemos importar archivos en una colección, y manipularlos de la misma manera que los creadores de música MIDI manipulan sonidos sintetizados. También podemos comprimir los datos de los archivos de sonido para reducir su espacio en el disco duro y aumentar la transferencia de datos.

Active X, Animaciones, CGI, Correo electrónico, Diseño gráfico, FTP, FrontPage, Html, Html Dinámico, Java, Java Script, Navegadores, Visual J ++.

PREPÁRATE PARA SER EL MEJOR WEBMASTER



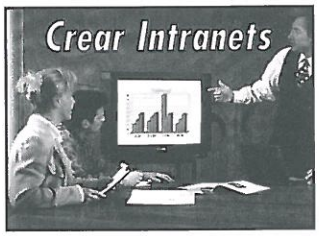
DESCRIPCIÓN DE LA COLECCIÓN:

- 5 CD ROM. Con los programas necesarios para el seguimiento del curso y el código de los ejercicios expuestos en el mismo.
- 500 fichas a todo color.
- 4 carpetas rígidas con anillas.
- 1 carpeta para los disquetes y CD-Roms.
- 48 disquetes.

Contenidos prácticos con ejemplos, ejercicios, casos, modelos, guías, etc... Se entregará un diploma acreditativo a las personas que finalicen el curso.

Aprenderás a:

Valorado en 22.270 Ptas.



curso WEBMASTER de programación Internet

Por sólo 29.990 PTAS

Agora também em português
Informação tel.: 01-8618090 em Lisboa

Solicítanos información en el tfno. 91-6614211, en el correo electrónico suscrip@towercom.es o envíanos este cupón y te contaremos cómo suscribirte y cómo funcionará la primera Bolsa de trabajo de expertos en Internet relacionada con este Curso.

Nombre y Apellidos ..... F. nacimiento .....
Domicilio ..... C.P. .....
Ciudad ..... Provincia ..... Telf.: .....
e-mail ..... Profesión .....
Apartado de correos ..... Código Postal .....

☐ Deseo recibir amplia información sobre el Curso Webmaster

Enviar a: TOWER COMMUNICATIONS c/Aragoneses, 7. 28.108 Alcobendas (MADRID)

VISUAL BASIC

Creación de controles ActiveX (II)

Jordi Agost (agost@eup.udl.es)

En el artículo anterior vimos las bases que necesitábamos para crear nuestro propio control ActiveX. En ese primer ejemplo fuimos capaces de crear un control ActiveX, pero quedaron en el aire muchos temas.

En un proyecto podremos incluir tantos módulos *UserControl* como deseemos, en una gran parte de los tipos de proyectos de Visual Basic (simplemente se definirá el módulo y éste se guardará en un archivo **.CTL**). Pero si lo que se quiere es suministrar controles a terceros o a otras aplicaciones entonces sólo lo podremos hacer cuando se crea un proyecto de control ActiveX, de tal forma que se generarán los correspondientes archivos **.OCX**.

Estos archivos serán los que se entreguen a terceros en el momento de la distribución del producto. Para todos los demás tipos de proyectos, dichos controles serán privados (es decir, sólo accesibles desde el mismo proyecto). Si posteriormente deseamos convertir dichos controles privados en públicos sólo tendremos que copiar los archivos **.CTL** a un proyecto de tipo control ActiveX y compilarlo para crear un control de estas características (un fichero con extensión **.OCX**), que podremos distribuir.

Si deseamos que nuestro control sea público, es decir que lo podamos utilizar en cualquier aplicación que desarrollemos únicamente tendremos que establecer al valor *True* la propiedad *Public* del objeto *UserControl*. Acto seguido lo compilaremos, y ya lo podremos utilizar en otra aplicación (siempre y cuando tengamos el archivo **.OCX** correspondiente).

Podría ocurrir que el proyecto estuviera marcado para ejecución desatendida (que indica que el proyecto está pensado para su ejecución sin ninguna interacción por parte del usuario), con lo que no contendrá ningún elemento de interfaz. En este caso, cualquier función que genere mensajes en tiempo de ejecución, con los que se realiza la interacción con el usuario, se escribirán en un registro de eventos. Si esto es así, no podremos incluir ningún objeto *UserControl* en él, ya que éste necesariamente posee una interfaz con el usuario.

El tipo de control que desarrollemos afectará de forma notable a la distribución de nuestra aplicación

Otra excepción sería el hecho de tener un control que sólo quisiéramos utilizar como constituyente de otros controles, ya que entonces marcaríamos su propiedad *Public* como **False**, con lo que el control sólo estaría disponible para los controles que lo contengan.

Es importante tener claras las ideas sobre el tipo de control que deseamos (privado o público) ya que esto afectará de forma notable a la distribución de nuestra aplicación. Si elegimos un con-

trol privado, su código se incorporará al ejecutable que tengamos, con lo que ahorraremos la distribución de un archivo **.OCX** adicional. Además, la depuración del control es más sencilla, pues no tendremos que probar todos los posibles casos sino únicamente aquellos que utilice la aplicación. Asimismo, como la compilación genera un único ejecutable, no tendremos que preocuparnos de si la aplicación funcionará con futuras versiones del control o no.

Por el contrario, si al cabo de un tiempo encontramos un fallo en el control tendremos que recompilar toda la aplicación, cuando de la otra forma hubiese bastado con redistribuir el archivo **.OCX**. Otro inconveniente es el espacio en disco, ya que si dos o más aplicaciones comparten el mismo control, al estar éste dentro del ejecutable se está duplicando código.

La aplicación genera un único ejecutable, no deberemos preocuparnos de versiones posteriores

Además, a largo plazo puede ser, el mantenimiento del control resultará más complicado. Es mucho más fácil mantener un control si éste se encuentra compilado aparte que si se encuentra dentro de la misma aplicación, si esto sucede, podemos tener el peligro de poseer en un momento dado diversas versiones del mismo control, con el consecuente peligro que esto conlleva.

INTERACCIÓN CON EL CONTENEDOR

Tal y como ya avanzamos en el artículo pasado una instancia de un control nunca existirá por sí misma, sino que siempre se debe colocar en un objeto contenedor. En el ejemplo del artículo anterior, dicho objeto contenedor lo constituía un formulario de Visual Basic. Estos objetos contenedores proporcionan a su vez métodos, propiedades y eventos adicionales, que al usuario le parecerán que forman parte del mismo control. Asimismo también podemos recopilar información del contenedor, que nos puede servir como referencia para establecer las propiedades propias del control. Veamos en primer lugar la forma de acceder a dichas propiedades.

Tabla 1. Propiedades del control

Nombre	*Tipo*	*Descripción*
Name	String	Contiene el nombre del control definido por el usuario.
Visible	Boolean	Especifica si el control es visible o no.
Parent	Object	Objeto de sólo lectura que representa el contenedor del control.
Cancel	Boolean	Indica que el control es el botón Cancel predeterminado del contenedor
Default	Boolean	Indica que el control es el botón predeterminado del contenedor.

La propiedad UserControl se establece por el programador que utilice luego el control

Cuando creamos una instancia de nuestro control, en la ventana de propiedades, aparecen algunas que no hemos creado nosotros. A todas estas propiedades se les llama extendidas y provienen del contenedor en el cual se encuentra el control, aunque parecen una parte del mismo. Ejemplos de propiedades extendidas serían: *Name*, *Tag* y *Left*. Un objeto de tipo *UserControl* tendrá acceso a las propiedades extendidas a través del objeto *Extender*. Para ver dicho funcionamiento utilizaremos el ejemplo del artículo anterior o creamos un nuevo proyecto de tipo control ActiveX . Le agregamos una etiqueta que será la que constituirá el título de nuestro control. Crearemos una propiedad nueva llamada **Titulo** (doble clic en él para visualizar el código, menú **Herramientas** y después **Agregar procedimiento de tipo propiedad**) y añadimos el siguiente código:

```
Public Property Get Titulo() As String
   Titulo = LabelPrueba.Caption
End Property

Public Property Let Titulo(ByVal NewTitulo As
      String)
   LabelPrueba.Caption = NewTitulo
   PropertyChanged "Titulo"
End Property
```

Para inicializar la propiedad **Titulo** del control que acabamos de crear, tendríamos que efectuar lo siguiente:

```
Private Sub UserControl_InitProperties()
   lblTitulo = Extender.Name
End Sub
```

Normalmente las propiedades extendidas del objeto *UserControl* se establecerán por parte del programador que utilice luego el control. Por ejemplo es absurdo que nosotros intentemos establecer la propiedad *Top*, ya que es el programador el que elige el sitio donde controla el control dentro del contenedor. Fijémonos que hemos colocado el código en el evento *InitProperties*, si hubiésemos escrito el siguiente código:

```
Private Sub UserControl_Initialize()
   lblTitulo = Extender.Name
End Sub
```

Hubiera aparecido un error, ya que el objeto *Extender* no se encontrará disponible hasta que haya ubicado el control en

Tabla 2. Propiedades de los contenedores

Nombre	*Tipo*	*Descripción*
Container	Object	Representa el contenedor del control.
DragIcon	Image	Especifica el icono exacto que se utiliza cuando se arrastra el control.
DragMode	Integer	Especifica si el control se arrastrará automáticamente o si el usuario del control deberá llamar al método **Drag**.
Enabled	Boolean	Especifica si el control está habilitado. No está presente si el control no tiene una propiedad **Enabled** con el identificador del procedimiento correcto.
Height	Integer	Especifica el alto del control en unidades de escala del contenedor.
HelpContextID	Integer	Especifica el identificador de contexto que se utiliza si se presiona la tecla **F1** cuando el control tiene el foco.
Index	Integer	Especifica la ubicación que ocupa esta instancia del control dentro de una matriz de controles.
Left	Integer	Especifica la ubicación del borde izquierdo del control respecto al borde izquierdo del contenedor.
TabIndex	Integer	Especifica la ubicación del control en el orden de tabulación de los controles del contenedor.
TabStop	Booleano	Especifica si el tabulador se detendrá o no en este control
Tag	String	Contiene un valor definido por el usuario.
ToolTipText	String	Contiene el texto que aparece cuando el cursor se detiene sobre el control durante más de un segundo.
Top	Integer	Especifica la ubicación del borde superior del control respecto al borde superior del contenedor.
WhatThisHelpID	Integer	Especifica el identificador de contexto exacto que se emplea cuando se utiliza el cuadro emergente "Qué es esto" en el control.
Width	Integer	Indica tamaño de ancho del control en las unidades de escala del contenedor.

el contenedor, por lo que debemos evitar cualquier código en el evento *Initialize* (debemos esperar a los eventos *InitProperties* o *ReadProperties*).

Algunas propiedades de extensión son en principio estándar, mientras que otras son específicas para cada contenedor. Si en algún momento accedemos a alguna de estas propiedades no estándares del contenedor, puede suceder que el control que estamos diseñando pase a ser específico para dicho contenedor (a no ser que incluyamos control de errores en el código). Además en el momento de compilar el control, Visual Basic no tiene ninguna forma de conocer las propiedades extendidas que estarán disponibles cuando se ejecute el control, por lo que las referencias a las propiedades extendidas siempre se enlazarán en tiempo de ejecución. Las propiedades estándar de cada contenedor son las que se especifican en la tabla 2.

La mayor parte de contenedores también proporcionan propiedades extendidas adicionales como *Left*, *Top*. *Width* y *Height*.

El contenedor posee lo que se denominan Propiedades de ambiente

Hasta ahora hemos visto las propiedades adicionales o extendidas que proporciona el contenedor. A partir de ahora estudiaremos la información que puede suministrar el contenedor. El contenedor posee lo que se denomina *Propiedades de Ambiente*, que informan al control de cuestiones tales como las mejores formas de mostrarse en el contenedor. Se accede a dichas propiedades a través del objeto *AmbientProperties*, el cual se distingue del objeto *Extender* por contemplar todas las propiedades estándar, tanto si el contenedor las incorpora como si no (en éste último caso devolverá las propiedades indicadas por defecto en la ayuda).

De esta forma ya no necesitamos incorporar código de control de errores, si las propiedades se encuentran visibles en el examinador de objetos. En otro caso significará que son características propias del contenedor y sí debemos hacer un control de errores. Existen muchas propiedades de ambiente, pero vale la pena tener en cuenta algunas de ellas, como por ejemplo:

En el momento de compilar el control, Visual Basic no tiene ninguna forma de conocer las propiedades extendidas disponibles

UserMode: permitirá a una instancia del control saber si se ejecuta en tiempo de diseño (valor False) o de ejecución.

DisplayName: podemos incluir el valor de esta propiedad en el tratamiento de errores para saber la instancia que ha producido el error.

ForeColor, BackColor, Font, TextAlign: son propiedades que el control puede adquirir de su contenedor para tener un aspecto semejante.

Tabla 3. Métodos

Método	*Descripción*
Drag	Inicia, termina o cancela una operación de arrastre del control.
Move	Mueve la ubicación del control
SetFocus	Establece el enfoque dentro del control
ShowWhatsThis	Presenta un tema seleccionado de un archivo de ayuda mediante el cuadro emergente "Qué es esto" proporcionado por la ayuda.
ZOrder	Coloca el control al principio o al final del eje Z dentro de su propionivel gráfico.

Tabla 4. Eventos

Evento	Descripción
DragDrop	Se desencadena cuando otro control del formulario se coloca en este control.
DragOver	Se desencadena cuando otro control del formulario se arrastra sobre este control.
GotFocus	Se desencadena cuando este control recibe el foco.
LostFocus	Se desencadena cuando este control pierde el foco.

También se debe destacar el evento *AmbientChanged*, que se produce si la apariencia o comportamiento del control se ve afectado por los cambios en alguna de las propiedades del objeto *AmbientProperties*. El argumento es una cadena con el nombre de la propiedad que ha cambiado.

NUESTRO CONTROL COMO CONTENEDOR DE OTROS

Si lo que deseamos es que los programadores puedan incorporar otros controles (al igual que ocurre por ejemplo con el control *PictureBox* de Visual Basic), tenemos que establecer a *True* la propiedad *ControlContainer* del objeto *UserControl*. Al incorporar controles dentro del nuestro, Visual Basic creará la colección *ContainedControls* dentro del objeto *UserControl*. Dicha colección será accesible en tiempo de diseño y en tiempo de ejecución.

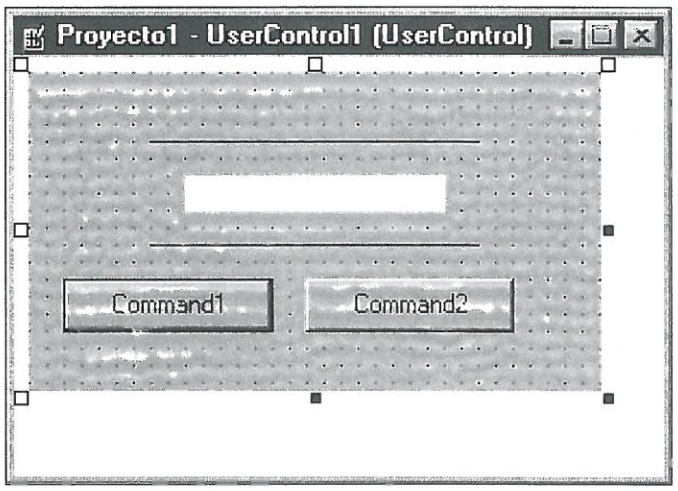
El argumento es una cadena con el nombre de la propiedad que ha combiado

Dicho procedimiento no funcionará en todos los contenedores, puesto que para que funcione en concreto, dicho contenedor debe poseer o admitir una interfaz del tipo *IsimpleFrame*, que permite habilitar la característica *ControlContainer*. Además el contenedor deberá implementar una interfaz de tipo *IVBGetControls*, ya que si no las llamadas que se hagan a la colección *ContainedErrors* pueden producir error, con lo que nos veremos obligados a escribir código para controlar los posibles errores.

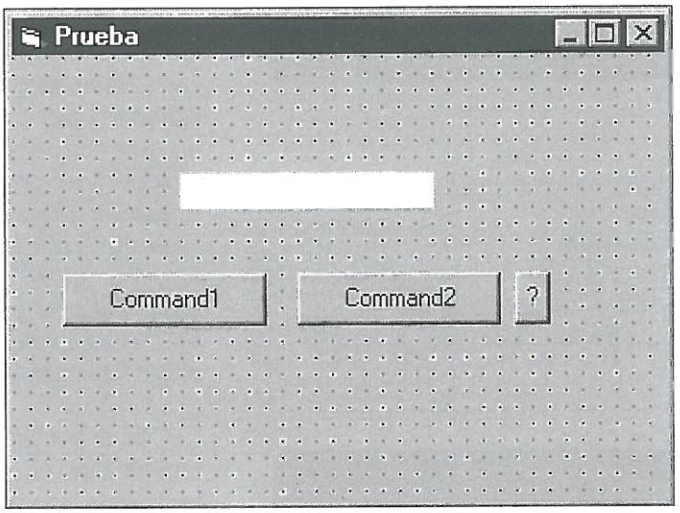
Si el contenedor no admite esta característica, el control funcionará perfectamente excepto cuando se desee incorporar un control dentro de él, cosa que no va a ser posible.

Si deseamos que los programadores puedan incorporar otros controles debemos establecer a True la propiedad ControlContaineer

Es importante diferenciar la colección *Controls* de la *ContainedControls*, ya que la primera sólo tendrá los controles constituyentes utilizados al diseñar nuestro control ActiveX. Imaginemos que tenemos un control como el siguiente:



Control diseñado por nosotros.



Al mismo control un programador le ha añadido un tercer botón de ayuda.

Luego un hipotético programador le añade un tercer botón para la ayuda.

Acto seguido ejecutamos en el control los siguientes códigos:

```
Dim obj As Object

For Each obj in UserControl.Controls
   MsgBox obj.Name
Next
```

Aparecerán los nombres de los controles constituyentes: *Command1, Command2, Line1, Line2, lblTitulo*

```
Dim obj As Object

For Each obj in UserControl.ContainedControls
   MsgBox obj.Name
Next
```

Ahora solamente aparece el nombre: *CommandHelp* y además sólo para esta instancia.

Para que la propiedad Enabled funcione correctamente, tendremos que asignarle el identificador de procedimiento de Enabled

La colección *ContainedControls* no se encuentra disponible en el evento *Initialize* del objeto *UserControl*. No estará disponible hasta que el control se ubique dentro del contenedor, y esto sucederá cuando el objeto *UserControl* reciba el evento *ReadProperties*. Además en el momento de diseñar un control de tipo contenedor debemos tener en cuenta que este hecho afectará negativamente al rendimiento del sistema, ya que estamos añadiendo una nueva capa a nuestro control.

Un último aspecto al que se debe prestar atención cuando se habla de controles contenedores es que si nuestro control contenedor posee un fondo transparente y un programador coloca un control sobre el fondo transparente, éste no se va a ver, por lo que es conveniente no tener fondos transparentes si nuestro control va a actuar como contenedor.

HABILITAR O DESHABILITAR

La habilitación de un control se realiza mediante la propiedad *Enabled*. Se trata de una propiedad extendida, pero que el objeto *Extender* la proporciona, a no ser que nuestro control la posea y además el identificador del procedimiento sea el correcto. Si dicha propiedad no está presente, el control no tendrá el mismo comportamiento (en lo que a su habilitación se refiere) que otros.

Creamos la propiedad y dejamos que esta ejerza su influencia sobre la propiedad *Enabled* del objeto *UserControl*:

```
Public Property Get Enabled() As Boolean
   Enabled = UserControl.Enabled
End Property
Public Property Let Enabled(ByVal NewValue As
      Boolean)
   UserControl.Enabled = NewValue
   PropertyChanged "Enabled"
End Property
```

Además para que la propiedad *Enabled* funcione correctamente, tendremos que asignarle el identificador de procedimiento de **Enabled**, que permitirá elegir un identificador estándar para la propiedad, el evento o el método que previamente hayamos seleccionado. El identificador lo utilizaremos para identificar una propiedad, un método o un evento como un tipo estándar que puede que conozcan las aplicaciones *host* del control.

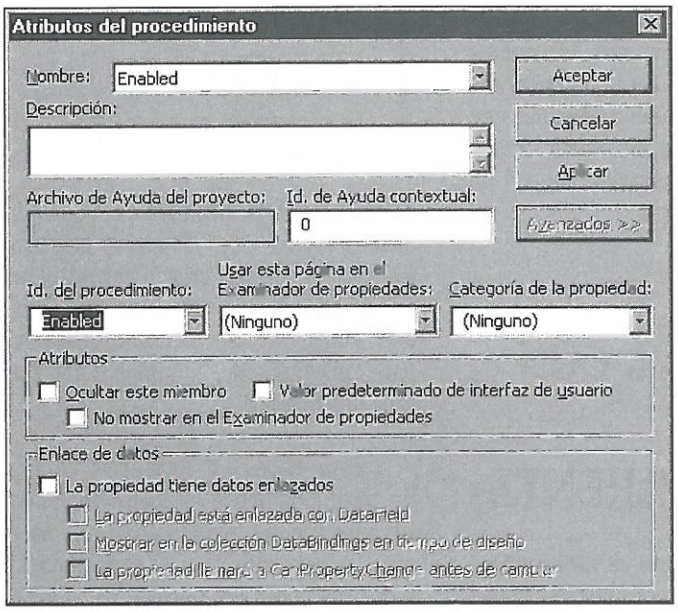
La habilitación de un control se realiza mediante la propiedad Enabled, que es de tipo extendida

Para asignar el identificador de procedimiento de la propiedad *Enabled* accedemos al menú **Herramientas**, y a continuación a la opción **Atributos del procedimiento** (ver figura, de los atributos de procedimientos).

En el cuadro *Nombre* seleccionamos el procedimiento *Enabled*, seguidamente hacemos un clic en el botón **Avanzadas**, y acto seguido en el cuadro de texto *Id*. Dentro de los procedimientos seleccionamos *Enabled* para dar a la propiedad el identificador correcto. Ahora, al haber elegido el

identificador de procedimiento correcto, el objeto *Extender* la sustituirá por su propia propiedad *Enabled*, así cuando el usuario establezca la propiedad extendida, el contenedor se encargará de establecer la propiedad que nosotros hayamos creado.

Al igual que antes con las propiedades de apariencia, debemos evitar establecer la propiedad *Enabled* del objeto *UserControl* (si exceptuamos la propiedad *Enabled* del mismo control). Esto sucede así porque es el contenedor el encargado de habilitar y deshabilitar los controles que conllleva.



Pantalla de atributos de procedimientos donde asignamos los identificadores de procedimientos de la propiedad Enabled.

BOTONES PREDETERMINADOS

Si lo que queremos es que nuestro control sea un botón predeterminado, o en su defecto un botón de cancelar, debemos establecer a *True* la propiedad *DefaultCancel* del objeto *UserControl*, ya que si no se produce de esta manera estos botones pasarán a estar controlados por el contenedor. Si deseamos saber si tenemos que dibujar una pequeña raya indicando que el control es el seleccionado por defecto, entonces podemos consultar la propiedad *DisplayAsDefault* del objeto *AmbientProperties*. Si como sucede muy a menudo el usuario presiona la tecla **Enter** para dar por válida la opción predeterminada, entonces se producirá el evento *AccessKeyPressed* del objeto *UserControl*.

Además, si vamos a programar un control ActiveX, vale la pena considerar si éste pudiese ser sustituido por una DLL ActiveX

Vamos ahora a tratar temas un poco "más gráficos", es decir, referidos al aspecto visual del control y a su apariencia por pantalla.

HACER EL CONTROL INVISIBLE EN TIEMPO DE EJECUCIÓN

No podemos utilizar la propiedad *Visible* del objeto *Extender* para hacer invisible el control en tiempo de ejecución, ya que el usuario tendrá acceso a las propiedades extendidas, razón por la que podrá si así lo desea, hacer visible el control.

Además si vamos a programar un control ActiveX, vale la pena considerar si éste pudiese ser sustituido por una DLL ActiveX (siempre y cuando no utilicemos propiedades pertenecientes a los controles ActiveX), ya que éste último método de programación requiere muchos menos recursos y esto no puede provocar problemas.

Debemos, cuando sea posible, evitar establecer la propiedad Enabled del objeto UserControl

Asimismo si nos fijamos en algún ejemplo de control invisible en tiempo de ejecución, como por ejemplo el control *ActiveX Crystal Reports* incluido en Visual Basic (sirve para hacer informes) veremos que posee una tamaño fijo. Si queremos que nuestro control también posea un tamaño fijo lo único que tenemos que hacer es incluir el siguiente código:

```
Private Sub UserControl_Resize()
  Size 300, 300
End Sub
```

ALINEACIÓN

Si vamos a crear un control que necesite alinearse sobre el formulario (como por ejemplo pasa con el control de la barra de estado), tendremos que establecer a *True* la propiedad *Alignable* del objeto *UserControl*. Si nuestro contenedor permite la alineación de controles,

entonces se agregará la propiedad *Align* al objeto *Extender*, aunque deberemos tener en cuenta que habrá contenedores que no admitan controles alineables con lo que será necesario introducir código que nos haga un exhaustivo control de errores.

CONTROL CON FONDO TRANSPARENTE

Para que nuestro control se vuelva transparente, es decir que podamos ver lo que hay detrás de él tenemos que establecer la propiedad *BackStyle* del objeto *UserControl* a *Transparent*. Si dentro de nuestro control existe por ejemplo un control de tipo *Shape* y también su propiedad *BackStyle* es **Transparent** entonces los clics que se realicen sobre esta zona se pasarán al contenedor.

Si nuestro contenedor permite la alineación de controles, entonces se agregará la propiedad Align al objeto Extender

Si en vez de ser un control *Shape* es una caja de texto, un control *Label*, cuya su propiedad *BackStyle* tiene asignada un valor de **Transparent** entonces sólo procesaremos los clics que caigan exactamente encima de las letras. Si el control se trata ahora de un mapa de bits (que establecemos con la propiedad *MaskPicture* del control *UserControl*), su propiedad *BackStyle* también la estableceremos a **Transparent** y con la propiedad *MaskColor* podemos decirle cual es el color transparente. Todo lo que sea de este color pasará a ser transparente y los eventos del ratón se pasarán al contenedor.

AGREGAR UN CUADRO "ACERCA DE"

Normalmente cada control dispone de un cuadro "Acerca de..." donde se muestran los autores que han creado el control, así como el distribuidor del control o archivo .ocx. Podemos hacer un cuadro Acerca de para todos nuestros controles o uno para cada uno de ellos.

Para hacer un formulario Acerca de, añadimos uno al proyecto de control ActiveX (no al proyecto en el que hacemos las pruebas) y en él especificamos todos los controles necesarios, o en su defecto elegimos un tipo propuesto por Visual Basic, de controles predefinidos con lo que ya tendremos la estructura montada. Dejamos el nombre por defecto (frmAbout) y además creamos un procedimiento para mostrar el cuadro de diálogo.

```
Public Sub MostrarAbout()
  frmAbout.Show 1
  Unload frmAbout
  Set frmAbout = Nothing
End Sub
```

La última línea la utilizamos para liberar la memoria que hubiésemos podido llegar a emplear.

Visual Basic utilizará automáticamente el nombre de clase del control como texto de información sobre herramientas

Acto seguido en el menú **Herramientas**,debemos seleccionar **Atributos de procedimiento**, elegimos nuestro formulario y en el cuadro de *Identificador de procedimiento* seleccionamos **AboutBox**, paso con el que damos el identificador correcto a nuestro formulario Acerca de.

Si lo deseásemos solo bastaría con repetir este paso tantas veces como controles tengamos.

UN MAPA DE BITS PARA EL CUADRO DE HERRAMIENTAS

El tamaño de un mapa de bits del control *ToolBox* (caja de herramientas) es de 16 píxeles de ancho por 15 píxeles de alto. Si lo que deseamos es que nuestro control tenga una apariencia propia en la caja de herramientas todo lo que debemos hacer es crear un mapa de bits de este tamaño (recordemos que el mapa de bits puede ser de tipo **bmp, jpg, gif, dib o pal**) y luego asignarlo a la propiedad *ToolboxBitmap* de nuestro objeto *UserControl*.

Visual Basic utilizará automáticamente el nombre de clase del control como texto de información sobre herramientas y será el que aparecerá cuando el usuario ponga el cursor del ratón encima de dicho icono en la caja de herramientas.

Hasta aquí hemos visto temas que nos permiten profundizar más en el tratamiento de los controles.

Con las bases adquiridas en el pasado artículo y la ampliación de éste ya podemos comenzar a construir nuestros propios controles ActiveX con soltura, aunque aún nos quedarán algunos temas en los que profundizar, como por ejemplo la adaptación de un control a Internet.

Noticias frescas, cada día

TOWER
COMMUNICATIONS

EN INFORMÁTICA MARCAMOS EL CAMINO

NET MAGAZINE
PC MEDIA
SÓLO PROGRAMADORES
PC PLAYER

SIMULACIÓN DE SISTEMAS

Práctica con un sistema continuo

Antonio José Novillo López

Tras la introducción teórica realizada en el artículo anterior, ahora nos encontramos en condiciones de realizar nuestra primera simulación de un sistema real conocido por todos, la amortiguación de un coche, para ello aplicaremos una serie de conocimientos informáticos adicionales.

En el anterior número realizamos una introducción al mundo de la simulación de sistemas. En el presente artículo trataremos de exponer como se realizaría la simulación de un sistema dinámico mediante diversas herramientas, comenzando por C, un lenguaje de propósito general hasta finalizar en ACSL o Simulink, entornos específicos de simulación.

Dado el propósito didáctico del artículo el sistema que se simulará no será muy complicado. De este modo nos podremos fijar más en la metodología y olvidar un tanto el complejo desarrollo matemático de este tipo de sistemas. En este caso he escogido como sistema la amortiguación de un coche. Una vez conocido el sistema, el siguiente paso consiste en modelar este sistema mediante elementos que nos sean conocidos y que seamos capaces de describir matemáticamente.

En esta práctica, el modelo de una amortiguación de un coche se puede expresar como una masa unida a un muelle con un amortiguador en paralelo (ver figura 1). El objetivo de la simulación será determinar los valores óptimos de las constantes que definen tanto al muelle como al amortiguador para que la comodidad dentro del coche sea máxima dentro de unos límites de seguridad necesarios. De esta manera, realizaremos la simulación para un rango de valores de K y V de modo que, observando los resultados obtenidos seamos capaces de diseñar la amortiguación del auto. Únicamente con nuestra computadora, conseguiremos diseñar un sistema que de un modo tradicional hubiera necesitado complicados y caros mecanismos físicos.

El modelo matemático de este sistema sería el siguiente:

$$\frac{d^2 y(t)}{dt^1}+\left(\frac{v}{m}\right)\frac{dy(t)}{dt}+\left(\frac{k}{m}\right)yt=\left(\frac{v}{m}\right)\frac{dx(t)}{dt}+\left(\frac{k}{m}\right)y(t)$$

Donde m representa el valor de la masa, k la constante elástica del muelle y v la constante del amortiguador. La función $x(t)$ representa la forma que tendrá la carretera sobre la que avanza el coche. En general será una combinación de una función de baja frecuencia, que será la forma de la carretera, si está en cuesta, etc., y una función de alta frecuencia, que serán los baches que haya en el camino. En un caso real deberíamos conocer estos datos, normalmente como una función interpolada o mediante valores numéricos. En este artículo, para facilitar el cálculo utilizaremos una rampa, de modo que $x(t)=t$. Como condiciones de contorno supondremos que el amortiguador está parado en el instante inicial de tiempos:

$$\frac{dy(0)}{dt}=0$$

$$y(0)=0$$

Para realizar nuestro ejemplo daremos unos valores a las constantes un

tanto arbitrarios para facilitar los cálculos. En el caso de una simulación real, deberíamos tomar los datos reales de m, k y v. Los valores usados serán $m=0.5$, $k=1$ y $v=1.5$. Luego la ecuación a resolver será:

$$\frac{d^2y(t)}{dt^1}+3\frac{dy(t)}{dt}+2y(t)=3\frac{dx(t)}{dt}+2y(t)$$

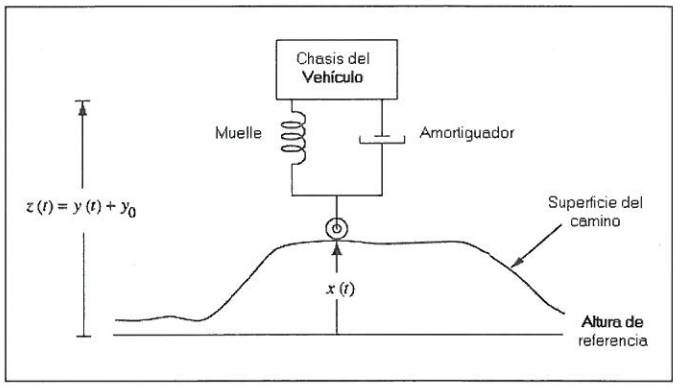
La solución general de esta ecuación con las condiciones de valor inicial indicadas anteriormente será de la forma:

$$y(t)=c_1e^{-2t}+c_2e^{-t}+a+bt$$

Esta información nos será útil para comparar los resultados de la simulación con el valor real y ver cuan acertado es nuestro método. La forma gráfica de esta solución para un valor de t grande será aproximadamente la de una rampa, puesto que el sistema trata de seguir la entrada, que en este caso es $x(t)$. Si llegamos a variar lo valores de las constantes obtendríamos otras soluciones que, en general, seguirán a la rampa pero provocarán oscilaciones que en nuestro sistema darán lugar a pequeños desplazamientos verticales del coche que disminuyen el confort de los ocupantes.

La amortiguación de un coche se puede simular mediante un muelle y un amortiguador

En los siguientes puntos se realizará la simulación mediante varias técnicas diferentes que nos permitirán ver los diferentes pros y contras de cada metodología. Para ello realizaremos la simulación en C, un lenguaje de propósito general muy adecuado para ello, en Matlab, una herramienta matemática, en ACSL, un lenguaje específico de simulación de sistemas y mediante Simulink, una toolbox de Matlab que permite simular sistemas de una forma casi visual y muy sencilla.



Modelo de la amortiguación de un vehículo.

SIMULACIÓN EN C

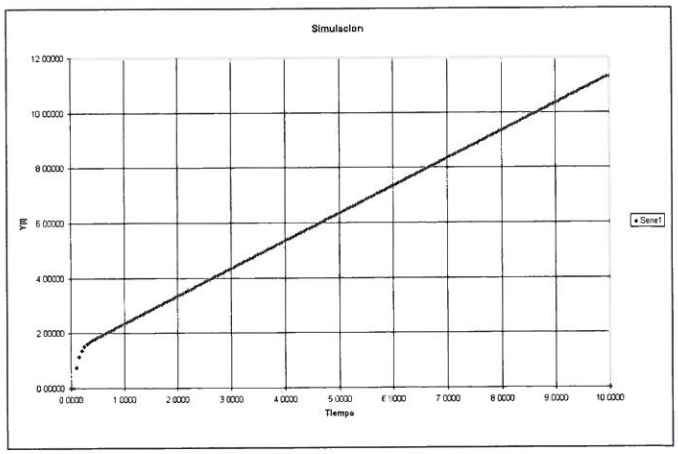
Se ha escogido C porque es uno de los lenguajes más extendidos por su enorme versatilidad, que permite al programador realizar todo tipo de programas. Esto a su vez es una desventaja para nuestro caso, que es un problema muy específico. Al no disponer de una herramienta especializada, deberemos resolver la ecuación diferencial numéricamente por nuestra cuenta. Esto es, deberemos implementar nosotros mismos un algoritmo de resolución de ecuaciones diferenciales.

Para ello, he utilizado el algoritmo de Euler, tras reducir la ecuación a un sistema de ecuaciones de primer orden, que es el método más sencillo. Para obtener un resultado mejor podríamos haber utilizado un algoritmo de Runge-Kutta de orden superior. No se va a entrar en detalle en cuanto al algoritmo porque es algo externo al tema central que se trata en el artículo. Para más información sobre el tema se aconseja revisar los títulos propuestos en la bibliografía.

Por otra parte, de cara a realizar simulaciones más serias utilizando este lenguaje, muy popular en entornos Unix, sería adecuado realizar o bien obtener alguna librería de cálculo numérico. De este modo, evitaríamos tener que realizar los algoritmos numéricos de resolución de las ecuaciones diferenciales. Tan sólo tendríamos que utilizar las funciones propias de las librerías y luego enlazar nuestro programa con ellas. En Internet se pueden encontrar algunas librerías de este tipo distribuidas libremente, aunque las que más abundan son las de Fortran, un lenguaje muy popular en entornos matemáticos y preferido por algunos profesionales hasta hace poco tiempo.

El código de la simulación será el siguiente:

```
#include<stdlib.h>
#include<stdio.h>

void main()
{
float t[200],y[200],x[200];
float k,h,c,m;
/* El incremento del tiempo será de 0.5 segundos
      */
float it=0.05;
int n;
FILE *fichero;

/* Abrimoe el fichero de salida */
fichero=fopen("salida.dat","w");

/* Introducimos los valores iniciales */
y[0]=0;
x[0]=0;
t[0]=0;
fprintf(fichero,"%f %f\n",t[0],y[0]);

/* Fijamos un valor de 0.5 para h que nos asegura
      la estabilidad */
h=0.5;

/* Bucle de resolución de la ecuación diferencial */

for (n=0;n<200;n++)
{
t[n+1]=(n+1)*it;
```



Resultado de la simulación usando el lenguaje C.

```
/* Resolvemos la ecuación diferencial mediante el
      método de Euler */
x[n+1]=x[n]+h*(3+2*t[n]-3*x[n]-2*y[n]);
y[n+1]=y[n]+h*x[n];

/* Guardamos los datos en un fichero */
fprintf(fichero,"%f %f\n",t[n+1],y[n+1]);
}

/* Cerramos el fichero de salida */
fclose(fichero);
}
```

Los resultados de salida, se obtienen en un fichero llamado **salida.dat**, que generará el programa durante su ejecución. Está formado por dos columnas. La primera contiene los valores del tiempo en segundos y la segunda el valor de *y(t)*, donde esta función representa la distancia desde la carrocería hasta la base de cotas. Si representamos gráficamente, mediante un programa exterior, por ejemplo, Excel, en la salida obtendremos la figura 2, que tiene una forma muy similar a la obtenida de una forma teórica.

SIMULACIÓN MEDIANTE MATLAB

El entorno Matlab trae herramientas que nos facilitarán mucho el trabajo frente a la programación tradicional. Matlab, para los que no lo conozcan, es un entorno orientado a los cálculos habituales en el álgebra matricial, que viene complementado con múltiples *toolboxs*, que son librerías que contienen funciones adicionales que permitirán resolver problemas más específicos, como pueden ser el control de sistemas, tratamiento de imágenes o lógica difusa (*fuzzy logic*).

Entre las ventajas de este entorno encontraremos que contiene funciones que resuelven directamente sistemas de ecuaciones diferenciales. De este modo, podremos obviar el algoritmo de resolución de la ecuación y centrarnos en nuestra simulación. Además, posee una estructura de programación bastante similar al C, lo que nos permitirá dominarlo con mayor facilidad.

En primer lugar debemos crear un fichero que describa nuestra ecuación diferencial. Se llamará *edo.m* y será de la forma:

```
function xdot=edo(t,x)
xdot(1)=x(2);
xdos(2)=3+2*t-3*x(2)-2*x(1);
```

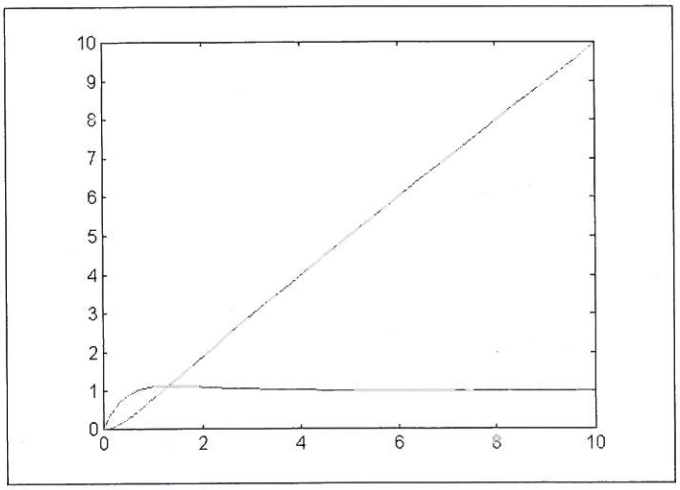
Seguidamente crearemos otro fichero llamado *simul.m* que contendrá el proceso de simulación completo:

```
t0=0; % Tiempo inicial de la simulación
tf=10; % Tiempo final de la simulación
x0=[0 0]; %Valores iniciales de la ecuación diferencial
[t,y]=ode45('edo',t0,tf,x0); %Solución numérica de la ecuación
plot(t,y)
```

La función *ode45* devolverá un vector *t* con el tiempo y una matriz *y* con dos columnas, una en la que aparece el valor *de y* junto con el valor *de*, la derivada de *y*. Otra ventaja que se puede observar en este entorno es que posee una potente función interna de representación gráfica para los resultados, evitando de este modo la necesidad de utilizar un programa externo como ocurrió con la simulación en C.

SIMULACIÓN EN ACSL

ACSL es un metalenguaje creado y diseñado específicamente para la simulación de sistemas continuos. Sus siglas provienen de *Advanced Continuous*



Resultado de la simulación utilizando Matlab.

Simulation Language. Con él podremos simular de una forma sencilla los más completos sistemas físicos. Además, como se apoya en el lenguaje Fortran, podremos crear nuestras funciones y librerías en este lenguaje en el caso de que necesitemos alguna función que el código no traiga implementada.

Tanto Matlab como ACSL poseen potentes herramientas que nos facilitarán la simulación

En la simulación mediante ACSL tendremos que dividir nuestro trabajo en al menos dos programas. En uno introduciremos el modelo de nuestro sistema, con sus ecuaciones y en otro los comando que se ejecutará en la simulación. En el caso de utilizar rutinas en Fortran o librerías deberíamos también incluir estos programas. El intérprete de ACSL se creará un programa en Fortran a partir de nuestros programas y lo compilará generando un ejecutable.

Nuestro modelo (**simul.csl**) se describirá como sigue:

```
PROGRAM simul
      DERIVATIVE
       CINTERVAL CINT=0.02
CONSTANT XIC=0.0 , XDIC=0.0, TSTP=10-0
TIME=INTEG(1.0,0.0)
XDD=3+2*RAMP(0)-3*XD-2*X
XD=INTEG(XDD,XDIC)
X=INTEG(XD,XIC)
TERM(T.GE.TSTP)
END
END
```

Aquí la definición es bastante diferente a la presentada en los métodos anteriores. Este lenguaje posee una potente función llamada *INTEG* que será capaz de integrar la ecuación diferencial dándole un valor inicial. De este modo *XDD* representa a la derivada segunda y *XD* la derivada primera.

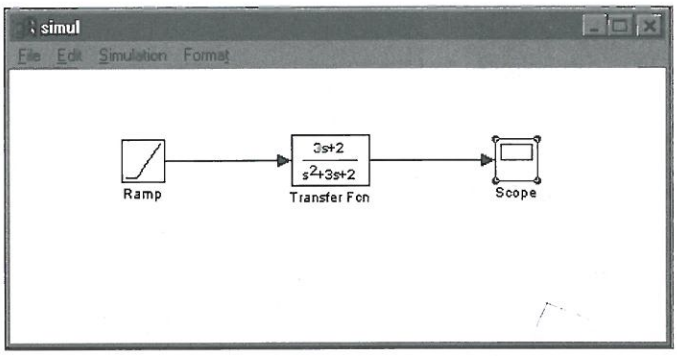


Diagrama de bloques del modelo en Simulink.

En segundo lugar definimos el fichero de comandos (**simul.cmd**):

```
SET TITLE="Simulación de la amortiguación de
      un vehículo"
SPARE
SET TCWPRN=72
OUTPUT TIME,XDD,XD,X,"NCIOUT"=20
PREPAR XDD,XD,TIME,X
START
RANGE "ALL"
PLOT "XAXIS" = TIME
PLOT X,XD
PLOT X,"HI"=1.0,"LO"=0.0, XD
PLOT "XAXIS"=XD, X
SPARE
STOP
```

Mediante estos comandos hemos configurado la simulación. Antes de **Start** hemos definido el formato que tendrá el fichero de salida (**simul.out**), donde encontraremos los valores numéricos de la simulación. Tras **Start** le hemos indicado al metalenguaje ACSL los gráficos que queremos que realice. En función de la versión de ACSL que utilicemos y si tenemos un posprocesador, estos gráficos serán en modo texto o bien en modo gráfico.

SIMULACIÓN MEDIANTE SIMULINK

Simulink es una *toolbox* de Matlab, por lo que requiere que esté instalado este programa para funcionar. En ella utilizaremos distintos bloques de entrada/salida para definir nuestro sistema. Dispondremos de distintos generadores de señal, varios tipos de gráficos de salida y bloques que describirán las partes de nuestro sistema.

La aproximación al problema aquí es un tanto diferente. Simulink es una herramienta utilizada habitualmente en el mundo de la Ingeniería de Control. Por ello, los sistemas se suelen describir de dos formas: o bien mediante ecuaciones de estado, o con funciones de transferencia. Para obtener la función de transferencia de un sistema deberemos tomar transformadas de Laplace en la ecuación diferencial que define el sistema.

Simulink permite realizar simulaciones de un modo casi visual

Al estar definido nuestro sistema por una única ecuación diferencial, será más fácil tomar transformadas y calcular la función de transferencia. En el caso de un sistema de ecuaciones diferenciales sería preferible, por comodidad, trabajar en variable de estado. De cualquier forma, existen bloques para definir nuestro sistema en cualquiera de los dos modelos.

Transformando nuestra ecuación obtendríamos la siguiente función de transferencia:

$$G(s)=\frac{Y(s)}{X(s)}=\frac{3s+2}{s^2+3s+2}$$

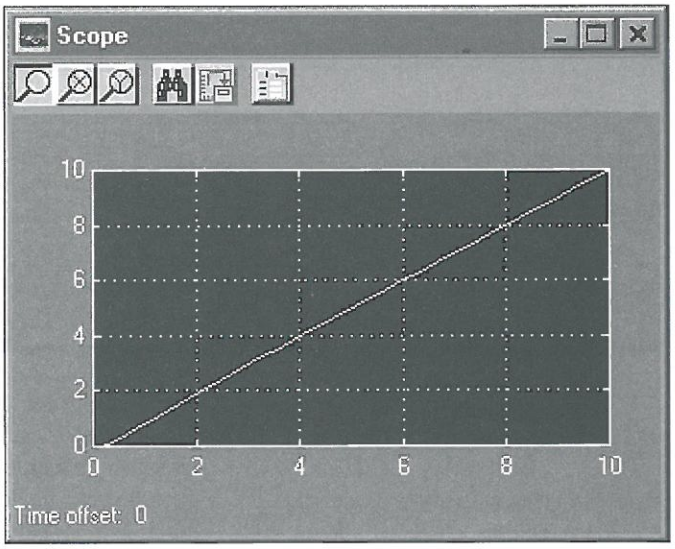


Diagrama de bloques del modelo en Simulink.

Donde Y(s) representa la señal de salida y X(s) la señal de entrada. En nuestro caso X(s) será una rampa. En el caso de querer utilizar una señal de entrada más complicada deberemos importarla desde un fichero. La posibilidad de importar ficheros como entrada nos permitirá utilizar modelos de terrenos mucho más complicados que en el caso de tener que describirlos analíticamente. En nuestro caso la forma del terreno se encuentra en el fichero simul.mat, por si el lector prefiere utilizar el fichero en lugar de la rampa para realizar el ejercicio.

Una vez tengamos configurada la entrada a nuestro sistema, utilizaremos un bloque de función de transferencia dentro del menú de sistemas lineales. Por último colocaremos un gráfico autoescalable como salida.

Ya solamente queda comenzar la simulación. En el menú de simulación encontraremos la opción Start. Dentro de este mismo menú encontraremos la opción Parameters, desde la cual podremos configurar los parámetros de nuestra simulación. Podremos escoger el periodo temporal de simulación, la precisión y el algoritmo utilizado para resolver las ecuaciones diferenciales.

La salida obtenida será de la forma que se muestra en la figura situada en la parte superior de esta página.

CONCLUSIONES

Como hemos podido observar, el proceso de simulación no es muy diferente de un entorno a otro. La gran diferencia aparece en forma de facilidad de resolución de nuestro modelo. Los entornos Matlab y ACSL contienen sus propias rutinas que facilitan enormemente el trabajo, además de incluir posprocesadores gráficos que permiten representar de una forma fácil y cómoda los resultados numéricos que hayamos obtenido tras la simulación. Sin embargo, la simulación no acaba aquí.

Posteriormente debemos estudiar estos resultados y alterar las variables del sistema hasta que obtengamos el resultado deseado. En nuestro caso vemos que la salida sigue casi perfecta a la entrada. Sería conveniente, y se aconseja al lector que lo realice como ejercicio adicional, cambiar los valores de k y v para observar como variaría la respuesta del sistema con valores diferentes. También es un ejercicio interesante cambiar la forma de la señal $x(t)$ y ver como respondería nuestra amortiguación a terrenos diferentes, como podría ser un terreno de forma senoidal, mucho más exigente para la amortiguación.

Es fundamental estudiar los resultados tras realizar la simulación numérica.

Resumiendo, podemos observar como el lenguaje C, pese a ser un lenguaje de gran versatilidad, es una solución complicada que obliga al usuario a realizar todas las tareas, algo que el resto de entornos facilita ofreciendo herramientas especializadas.

Sin embargo, Matlab no es el más adecuado para grandes simulaciones donde utilicemos sistemas muy complejos. En ese caso es donde emergen con fuerza las herramientas específicas como ACSL o Dymola, que es un entorno cuasi gráfico que genera una salida en ACSL. Por último reseñar la gran facilidad de manejo que ofrecen las herramientas gráficas tipo Simulink.

Muchas de las herramientas hasta ahora comentadas son de carácter comercial. Sin embargo, para algunas de ellas podemos encontrar opciones freeware o de libre distribución de igual calidad. Este es el caso de Matlab. Existe un programa clónico distribuido bajo licencia de GNU llamado Octave, que existe en versiones para Linux, FreeBSD y Windows 95.

La sintaxis de este programa es exactamente la misma que para Matlab y los programas desarrollados en Matlab serán ejecutables en Octave. Otra programa similar, aunque con distinta sintaxis es Scilab, un paquete de álgebra matricial desarrollado en Francia y que es gratuito para uso no comercial.

Es fundamental estudiar los resultados tras realizar la simulación numérica

Este paquete incluye una librería llamada Scicos que es muy similar a Simulink, permitiendo la simulación de sistemas, continuos e incluso híbridos, de un modo gráfico. En el caso de ACSL se pueden encontrar versiones para estudiantes a un precio más económico que el paquete profesional, algo que también ocurre con Matlab 5.0 y Simulink 2.0.

BUSCAMOS PROFESIONALES

Si te gusta escribir y tienes experiencia en algunos de estos campos de la informática

HARDWARE:

Tarjetas gráficas, modems, PC Cards, cámaras digitales, escáneres, tarjetas de sonido, impresoras, monitores, pantallas planas, dispositivos de almacenamiento, vídeo digital...

SOFTWARE:

Ofimática en general, aplicaciones profesionales de diseño, Internet, autoedición, multimedia, animaciones 3D, programación, sonido, edición de vídeo, antivirus, edición Web, juegos...

INTERNET:

Navegación, videoconferencia, conectividad, redes, ActiveX, *plug-ins*, Shockwave, RDSI, sistemas abiertos, comercio electrónico, sistemas de seguridad, intranet, extranet...

Envíanos tu C.V. a la siguiente dirección

Tower Communications
Apartado de correos 54.283
28080 - Madrid

(Ref.: Hard - Soft - Int)

PROGRAMACIÓN DE SOCKETS

Creación de un servidor Web con Delphi

Enrique de la Lastra (elastra@redestb.es)

Un servidor Web que proporcione el servicio básico de atender peticiones de páginas HTML es fácil de crear utilizando Delphi.
En este artículo iniciaremos la explicación para realizar uno propio.

INTRODUCCIÓN

En artículos anteriores correspondientes a esta sección hemos intentado internarnos y profundizar en la especificación de los *Windows Sockets*, que proporcionan la interfaz necesaria para la programación de aplicaciones de comunicaciones a través del protocolo TCP/IP. Con los conocimientos aprendidos, hemos creado un componente *socket* que encapsula las funciones básicas de dicha interfaz de forma que pueda convertirse en un *socket* cliente universal, sobre el que podemos programar nuestros propios componentes clientes y programas clientes de las aplicaciones estándares de Internet (así como de cualquier otra aplicación de comunicaciones de nuestra propia factoría).

En las siguientes líneas dejaremos de lado – de momento – el cliente *socket* que habíamos creado, y utilizaremos en el desarrollo de nuestro Servidor Web el componente *socket* servidor del "*Internet Solutions Pack*" (Paquete de soluciones de Internet) que viene incorporado en la versión Cliente/Servidor de Delphi 3. El componente aludido es el "TServerSocket" que se muestra, tal y como aparece en la paleta de componentes de Delphi, en la **Figura 1.**

Un servidor web (o servidor WWW) tiene, básicamente, la misión de atender las peticiones de páginas HTML que realizan los clientes. Esto supone enviarle al cliente no sólo los documentos HTML, sino todos los elementos que aparecen referenciados en él: imágenes, sonidos, vídeos, etc. Además el servidor web puede dialogar con el cliente gracias a las aplicaciones de servidor, que permiten que éste pueda, por ejemplo, enviarle datos al servidor y que en éste se actualice una base de datos o que se envíe al cliente una página HTML personalizada. Estas aplicaciones se pueden desarrollar bajo el estándar CGI (*Common Gateway Interface*, Interfaz de Pasarela Común), que permite el desarrollo de programas que dialogan con el servidor web, ampliando sus funciones. Si el servidor web empleado es el de Netscape o el de Microsoft, tenemos además la opción de crear aplicaciones de servidor mediante las interfaces NSAPI (*Netscape Server Application Programming Interface*) e ISAPI (*Internet Server Application Programming Interface*) respectivamente. Estas interfaces de programación permiten crear aplicaciones que dialoguen con los servidores web de ambas compañías (y que además sean más eficientes que las creadas a través de CGI).

El servidor web que nos proponemos crear proporcionará el servicio básico de distribución de páginas HTML a los clientes que las soliciten. Lo ampliaremos posteriormente para que permita el diálogo con los clientes Web, es decir para que proporcione soporte a CGI (por descontado no soportará NSAPI e ISAPI).

HTTP:EL PROTOCOLO DE LA WEB

Antes de comenzar a desarrollar, debemos conocer las particularidades del protocolo que sustenta la Web, por añadidura el protocolo que ha supuesto la revolución que hoy en día está introduciendo Internet en nuestra sociedad. Daremos unas breves pinceladas del mismo, pues un análisis detallado llevaría todo el artículo.

Una página hipertexto con imágenes o vídeos se denomina hypermedia

El protocolo de la Web es el *HTTP*, "*HyperText Transfert Protocol*" (Protocolo de Transferencia de HiperTexto). Nos paramos a analizar el significado de estas siglas. Por un lado se habla de "HiperTexto" y por otro de "Protocolo de Transferencia". El Hipertexto, se puede definir como un texto con formato, en el que pueden ir insertados objetos (imágenes, sonidos, vídeos) y que contiene Hiperenlaces, que a su vez se pueden definir como referencias dentro de un documento a otros documentos Hipertexto. El hecho de que en una página Hipertexto puedan ir insertadas imágenes o vídeos, amplía la definición del Hipertexto, que a veces se denomina Hipermedia. Teniendo en cuenta esto, el *HTTP* es el protocolo encargado de transferir documentos Hipertexto. Por otro lado, el lenguaje de Hipertexto más común – y el que de hecho se utiliza en Internet – es el HTML *(HyperText Markup Language)*

El protocolo HTTP especifica un mecanismo de transferencia de cualquier tipo de información a través de una red de comunicaciones. Tiene dos características destacadas: la primera es que todos

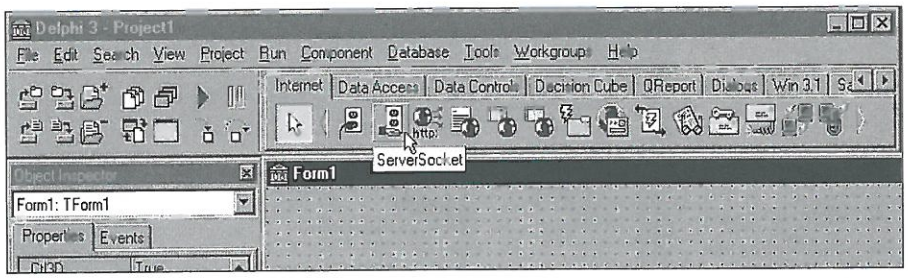

Delphi 3 - Project1
File Edit Search View Project Run Component Database Tools Workgroup Help
Internet Data Access Data Controls Decision Cube QReport Dialogs Win 3.1
ServerSocket
Form1
Form1: TForm1
Properties Events

Figura 1: Componente TServerSocket, en la pestaña Internet de la paleta de componentes.

los datos se envían como texto plano: tanto los comandos como el contenido aparecen como líneas de texto; y la segunda, que es un protocolo de aplicación y por tanto necesita de un protocolo por debajo para su transporte (que generalmente será el *TCP*, aunque en un principio puede ser cualquier otro). Cuando el protocolo de transporte es *TCP*, el *HTTP* utiliza como número de puerto el **80**.

Otras características del protocolo *HTTP* son:

- No conserva el "estado", es decir, para cada solicitud a través de HTTP se establece una conexión, se realiza la transferencia de la información y se libera la conexión. Por tanto, todas las peticiones realizadas a un servidor tienen que establecer y liberar una conexión, no pudiéndose beneficiar de las anteriormente establecidas (si no se suministra algún mecanismo adicional).

- No tiene necesidad de saber cómo manejar la información que transmite, ya que sólo se encarga de transferir los datos. Por ejemplo, no necesita saber cómo se muestra una imagen GIF para poder enviarla.

- Puede transferir cualquier tipo de información, además de páginas HTML: documentos, ejecutables, ficheros de sonido, imágenes, etc.

- La información está distribuida: uno puede crear un servidor de información con documentos residentes en muchas máquinas distintas, a las que se hace referencia con simples hiperenlaces.

Más concretamente y para el fin que a nosotros nos interesas, el protocolo *HTTP* define cómo un navegador (*browser*) envía peticiones y cómo el servidor web las responde. El navegador genera una petición cuando el usuario introduce una URL (*Uniform Resource Locator,* Localizador Uniforme de Recursos) o hace clic en algún Hiperenlace. Una URL determina la dirección completa de un documento dentro de la Web. Una URL consta de varias partes (algunas de ellas opcionales):

El HTTP define cómo un browser envía peticiones y cómo el servidor web la responde

1.- El nombre del protocolo seguido de "://" **(http://)**
2.- El nombre completo de un Servidor **(www.towercom.es)**
3.- El puerto TCP por el que se debe entrar al servidor **(:80)**;
4.- El directorio dentro del servidor **(/home/principal/)**
5.- Y por último el nombre del documento (home.htm)

La URL completa del ejemplo sería, por tanto:

http://www.towercom.es:80/home/principal/home.htm

Las tres últimas partes son opcionales, debido a que el puerto utilizado

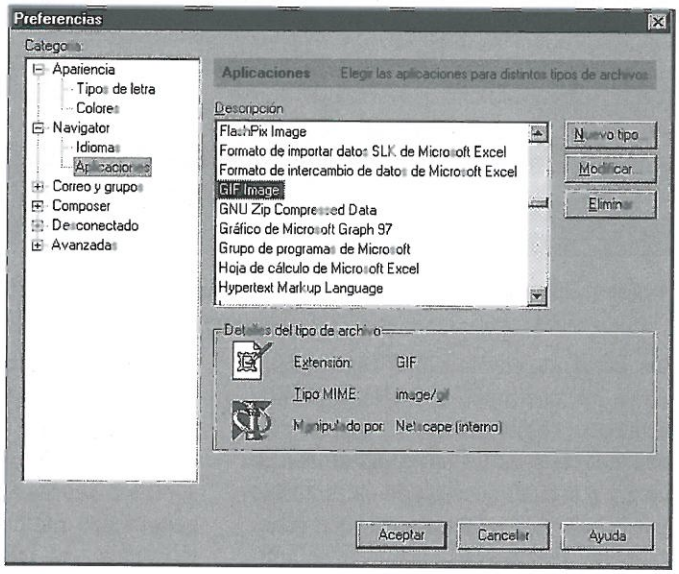


Figura 2: Opción Navigator/Aplicaciones del menú Edición/Preferencias de Netscape Communicator, desde la que se pueden consultar los tipos MIME soportados y se pueden especificar nuevos tipos.

suele ser el estándar (el **80** para *HTTP*) y a que el servidor especifica una página por defecto en caso de que no se introduzca ni la ruta ni el nombre del documento. Esta página suele ser: "**index.htm**" o "**home.htm**". Por otro lado, los navegadores modernos (Netscape, Microsoft), entienden que el protocolo utilizado por defecto es *HTTP*, y por tanto si sólo introducimos como URL: "www.ibm.com", añaden automáticamente "http://" delante.

PETICIONES Y RESPUESTAS HTTP

El diálogo entre un cliente web (como el navegador de Netscape o Microsoft) y un servidor web (como el que vamos a crear), se realiza a través de las peticiones que realiza el cliente y las correspondientes respuestas del lado servidor. Las peticiones del navegador componen un flujo de datos en forma de texto. La primera línea del texto enviado debe ser un comando de petición que normalmente será "*GET*" (conseguir) o "*POST*" (enviar), el nombre del URL y la versión del protocolo ("HTTP/1.0"). Las siguientes líneas contendrán información de cabecera y terminarán con una línea en blanco (necesaria según el estándar). Una petición típica, puede ser por ejemplo:

```
GET home/index.htm HTTP/1.0
accept: image/gif, image/jpeg, */*
user-agent: Mozilla/4.03 (Win95; I)
connection: Keep-Alive
(línea en blanco)
```

El servidor atenderá la petición y extraerá de la primera línea el nombre del recurso solicitado **home/index.htm** y en el caso de que sea un documento lo enviará al navegador. La información de cabecera *accept* (aceptar) indica que si una imagen está disponible en más de un formato, se preferirán el GIF y el JPEG. La cabecera *user-agent* (agente de usuario) identifica el navegador del cliente como Netscape Communicator 4.03 corriendo en Windows 95 sobre una máquina Intel. La cabecera *connection* (conexión) informa que el cliente obedece al protocolo **keep-alive** (manténte vivo), en el que la sesión establecida se reutiliza para siguientes peticiones. Otras cabeceras pueden definir el lenguaje preferido (en caso de que las páginas estén disponibles en varios idiomas), o suministrar una marca de tiempo, en la que se indica que el documento se recogerá de la caché del ordenador (si está en ella) salvo que exista una versión más moderna de la misma.

La cabecera connection (conexión) informa que el cliente obedece al protocolo keep-alive

El formato de la respuesta *HTTP* es similar al de la petición. Las líneas de cabecera terminan con una línea en blanco, a continuación de la cual se envían los datos del fichero solicitado. Una respuesta típica podría ser:

```
HTTP/1.0 200 OK
Server: Netscape-Communications/1.12
Last-modified: Monday, 14-Abr-98 08:55:00 GMT
Content-length: 8
Content-type: text/plain

¡Hola!
```

La respuesta comienza con una línea de estado y luego las líneas de cabecera, en texto plano. Después irá una línea en blanco (que indica el fin de las cabecera) y el resto de los datos, que pueden ser binarios.

Tabla 1: Los 3 métodos de petición soportados en la especificación HTTP/1.0

Método de petición	Función
GET	Pide toda la información solicitada en el recurso URL
HEAD	Pide sólo la información de la cabecera y ningún otro tipo de recurso
POST	Envía información al servidor web (por ejemplo un formulario insertado en una página web)

El número **200** que aparece en la primera línea nos informa de que nuestra petición ha tenido éxito. Los códigos **300** informan que no tenemos permisos para traernos el recurso solicitado. Códigos en el centenar **400** indican que el recurso solicitado no existe (porque se ha introducido erróneamente o porque ha cambiado de sitio); el más común es el **404: Not found.** Los códigos **500** suministran el error, si lo hay, de un servidor web.

Las dos siguientes líneas (2ª y 3ª) informan del tipo de servidor y de la última modificación de los datos que se envían.

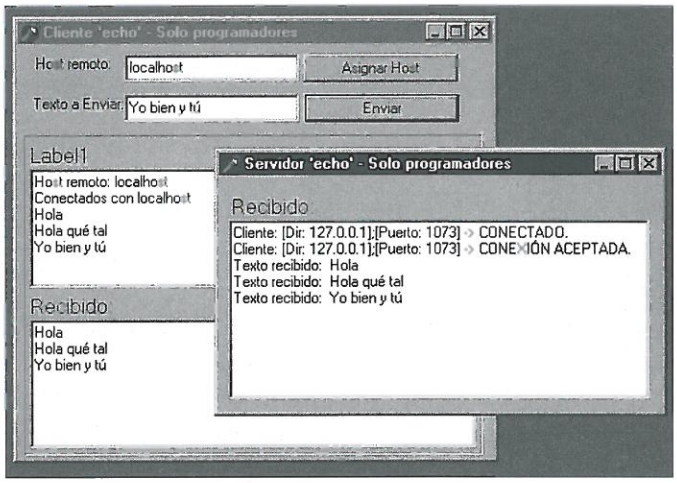


Figura 3: Cliente y Servidor del protocolo echo en acción.

Si el servidor suministra la cabecera *Content-length* (longitud de la cabecera) el navegador podrá saber -antes de recibir todo el recurso solicitado– cuántos bytes de datos se van a recibir.

El protocolo HTTP tiene hasta ahora tres versiones con implementaciones distintas

Existen extensiones de la versión 1.0 de HTTP que usarán esta información para mantener viva la sesión, de forma que las nuevas peticiones puedan viajar sobre ella. Si el servidor no puede suministrar la longitud de los datos, el navegador deberá recibir bytes hasta que el servidor termine la sesión -lo se indicará meidante *end of file* (fin de fichero) -. El problema será que entonces cada nueva petición del Navegador tendrá que abrir una nueva sesión con el servidor. Esto es más importante de lo que parece, ya que un documento HTML hace referencia a muchas imágenes, iconos, *applets* de Java, etc. Si la sesión se mantiene, todas estos "objetos" que van incrustados en la página HTML, se pueden transmitir en la sesión abierta. En caso contrario habrá que abrir una nueva sesión para cada objeto (¡a esperar más todavía!).

La cabecera *Content-type* (tipo de contenido) especifica el tipo MIME de los datos. Las siglas MIME responden a *"Multipurpose Internet Mail Extensions"* (Extensiones Multipropósito de Correo Internet), y son una extensión que se añadió inicialmente al correo electrónico para que se pudiesen enviar caracteres especiales (como acentos, eñes, diéresis, etc.) y para enviar cualquier tipo de documento como anexo de un correo electrónico, sin tener que codificarlo antes. Según MIME un tipo de datos se especifica como: "tipo de datos/subtipo de datos". De esta manera HTTP puede enviar cualquier tipo de datos: texto, imágenes, archivos ejecutables, clases Java, etc. En nuestro ejemplo: *text/plain*, indica que los datos vienen en formato de texto plano (esto es, sin formato). En la Figura 2, se muestra la ventana que aparece al seleccionar la opción "Preferencias..." en el menú *Edición* de Netscape Communicator. En la parte *Navigator/Aplicaciones*, podemos ver todos los tipos MIME predefinidos en el navegador y definir nuestros propios tipos MIME. En concreto, en la figura 2 está seleccionado el tipo de imagen GIF, cuyo tipo MIME es: *"image/gif"*. El tipo MIME más común es, cómo no, el HTML: *text/html*, que especifica un texto con formato de documento Web.

Tabla 2: Algunos de los encabezados de respuesta HTTP más comunes.

Encabezado	Función
Server	Indica el tipo de Servidor que emite la respuesta
Content-type	Indica, en formato MIME, el tipo de los datos enviados
Content-Encoding	Habilita el transporte a través de HTTP de otros datos que no tengan el formato ASCII
Content-Length	Informa del tamaño del recurso
Content-Version	Información de la versión de los datos
Last-Modified	Última fecha en que el recurso fue modificado
Expires	Indica cuándo el navegador debe volver a pedir los datos al servidor, en lugar de buscarlos en la *cache* local (si es que se encuentran allí)

TIPOS DE PETICIÓN Y RESPUESTA HTTP

El protocolo *HTTP* tiene, por ahora tres versiones, que por orden cronológico son: 0.9, 1.0 y 1.1. La versión 0.9 especificaba

Figura 4: Aspecto del Servidor de echo y propiedades modificadas en TServerSocket.

un tipo de petición denominado **Simple** y su correspondiente tipo de respuesta. Hoy día, ni los navegadores deberían generar peticiones simples ni los servidores web deberían generar respuestas de este tipo. Sin embargo, los servidores deben responder con respuestas simples si reciben del mismo tipo, con el fin de garantizar la compatibilidad hacia atrás. A partir de la versión 1.0 de *HTTP*, las peticiones y las respuestas se denominan **Completas** y son las que atenderá nuestro servidor web. Por tanto, en el resto del artículo nos olvidaremos de que existe la especificación *HTPP/0.9*, quedando pendiente para aquéllos que quieran que su servidor web sea aún más completo. De hecho, nuestro Servidor tampoco atenderá peticiones definidas de acuerdo a la norma *HTPP/1.1*; sólo hará caso de las peticiones *HTTP/1.0*.

Los navegadores no deberían generar peticiones simples y tampoco los servidores

Los mensajes de petición, como hemos visto en el apartado anterior, se componen de una o varias líneas, la primera de las cuales, deberá contener necesariamente el método de petición, el recurso solicitado y la versión del protocolo *HTTP*. En el ejemplo anterior esta línea era:

```
GET home/index.htm HTTP/1.0
```

Aquí el método es **GET**, el recurso **home/index.htm** y la versión del protocolo *HTTP*, la **1.0**. Los tres métodos de petición soportados por *HTTP/1.0* son: **GET**, **HEAD** y **POST** (Tabla 1). La norma *HTTP/1.1*, define 10 nuevos tipos de petición, adicionales a los tres anteriores.

Por su parte los mensajes de respuesta se compondrán de una línea de estado a la que siguen uno o varios encabezados (que son opcionales) y luego el cuerpo de la respuesta. Algunos de los encabezados más comunes se muestran en la Tabla 2.

COMPONENTE TSERVERSOCKET

El programa servidor hará uso del componente *TServerSocket*. Añadiendo este componente a un formulario de una aplicación Delphi, convertimos dicha aplicación en un servidor *TCP/IP*. Se encargará de permanecer a la escucha de peticiones de conexión y de aceptarlas llegado el caso.

Cada componente *TServerSocket* utiliza un objeto *TServerWinSocket*, que actúa como punto final para manejar las llamadas a la API Winsock en una conexión servidor que está "a la escucha". Este objeto se almacena en la propiedad *Socket*. Además *TServerSocket* utiliza un objeto *TServerClientWinSocket* para administrar las llamadas a la API Winsock para cada conexión con cada socket cliente. Es decir, se crea un objeto *TServerClientWinSocket* cuando un *socket* servidor acepta una conexión con un *socket* cliente.

Las propiedades más importantes de *TServerSocket*, además de la citada *Socket* son las siguientes:

- *Puerto*: Número de puerto *TCP* de la conexión. Habrá que fijarlo para poder escuchar las peticiones de clientes. En nuestro caso será el valor del puerto *TCP* para el protocolo *HTTP*: **80**.

- *Service*: Nombre del servicio para el que se utiliza la conexión. Esta propiedad es complementaria de la anterior, ya que la mayoría de los servicios de red tienen números de puerto estándar. Al especificarla (por ejemplo con el valor **ftp**, **http** ó **echo**), se fija el valor por defecto del puerto. Si se especifican ambas propiedades (puerto y servicio) se toma el valor de *Service*. Nuestro servidor especificará el valor **http**.

Los mensajes de respuesta tienen una línea de estado con uno o varios encabezados

- *Active*: ropiedad que abre o termina una conexión servidora. Si se fija a **True** en tiempo de diseño, se abre una conexión inmediatamente al ejecutar la aplicación. En tiempo de ejecución, se deben utilizar los métodos: *Open* (que pone a la escucha al *socket*) y *Close* (que hace que deje de escuchar, terminando todas las conexiones activas con aplicaciones clientes y cancelando las conexiones que estaban pendientes de aceptarse).

- *ServerType*: Tipo de servidor. Esta propiedad tiene dos posibles valores: **stNonBlocking** (modo de tra-

bajo asíncrono, en el que todas las conexione se tratan sobre un solo *thread* o hilo de ejecución) y **stThreadBlocking** (modo bloqueante, en el que cada conexión crea su propio *thread* bloqueante).

El evento OnAaccept ocurre cuando la conexión con el socket ha sido aceptada

Cuando un *socket* servidor está a la escucha (después de fijar *Active* a **True** o de llamar en tiempo de ejecución al método *Open*) acepta automáticamente las conexiones de los clientes que se van recibiendo. Para cada una, se genera un evento *OnClientConnect* ("al conectar con un cliente"). Además de este evento existen otros eventos útiles:

- *OnClientDisconnect*: Este evento se dispara cuando se cierra una conexión con un *socket* cliente.
- *OnClientRead*: Ocurre cuando el *socket* servidor tiene información disponible para recoger procedente de un *socket* cliente.
- *OnClientWrite*: Se dispara cuando el *socket* servidor debería enviar información a un *socket* cliente.
- *OnAccept*: Ocurre cuando la conexión con el *socket* cliente ha sido aceptada.
- *OnClientError*: Este evento ocurre cuando hay un fallo en el establecimiento, uso o terminación de una conexión con un *socket* cliente.

Una vez abierta una conexión, se puede obtener información de la misma a través del objeto *TServerWinSocket* que se crea con la conexión y que se almacena en la propiedad *Socket* de *TServerSocket*. Este objeto recopila

Listado 1: Código de la aplicación servidora del protocolo echo.

```
unit Unit_servidor_echo;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls,
  Forms, Dialogs,  ScktComp, StdCtrls, Winsock;

type
  TForm2 = class(TForm)
    ServerSocket1: TServerSocket;
    Memo1: TMemo;
    Label1: TLabel;
    procedure ServerSocket1ClientRead(Sender: TObject;
      Socket: TCustomWinSocket);
    procedure ServerSocket1Accept(Sender: TObject;
      Socket: TCustomWinSocket);
    procedure ServerSocket1ClientConnect(Sender: TObject;
      Socket: TCustomWinSocket);
  end;

var
  Form2: TForm2;

implementation

{$R *.DFM}

procedure TForm2.ServerSocket1ClientRead(Sender: TObject;
  Socket: TCustomWinSocket);
var Recibido: string;
begin
  Recibido := Socket.ReceiveText;
  Memo1.Lines.Add ('Texto recibido:  ' + Recibido);
  Socket.SendText (Recibido);
end;

procedure TForm2.ServerSocket1Accept(Sender: TObject;
  Socket: TCustomWinSocket);
begin
  Memo1.Lines.Add ('Cliente: [Dir: ' +
                StrPas(inet_ntoa(Socket.RemoteAddr.sin_addr)) +
                '];[Puerto: ' +
                IntToStr(Socket.RemotePort) +
                '] -> CONEXIÓN ACEPTADA.');
end;

procedure TForm2.ServerSocket1ClientConnect(Sender: TObject;
  Socket: TCustomWinSocket);
begin
  Memo1.Lines.Add ('Cliente: [Dir: ' +
                StrPas(inet_ntoa(Socket.RemoteAddr.sin_addr)) +
                '];[Puerto: ' +
                IntToStr(Socket.RemotePort) +
                '] -> CONECTADO.');
end;

end.
```

suscríbete

a Sólo Programadores

y consigue un magnífico descuento

suscripción **normal**

ahorro **20%**

12 revistas (1 año) por sólo...

9.350 ptas.

suscripción **estudiantes**
(carreras técnicas)

ahorro **40%**

12 revistas (1 año) por sólo...

7.050 ptas.

23
SÓLO PROGRAMADORES
EL PODER DE LOS 32 BITS
WATCOM C++
ESTUDIO DE LOS APPLETS DE JAVA
LINUX: CÓMO CREAR DISCOS DE EMERGENCIA A MEDIDA

números atrasados

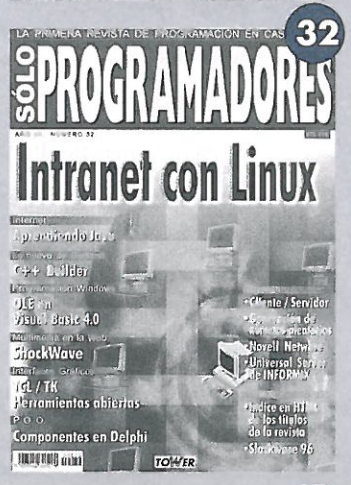
SÓLO PROGRAMADORES

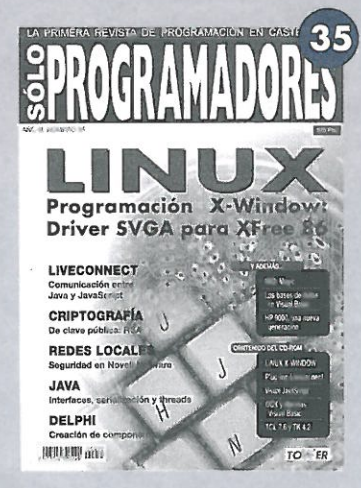
completa ya tu colección

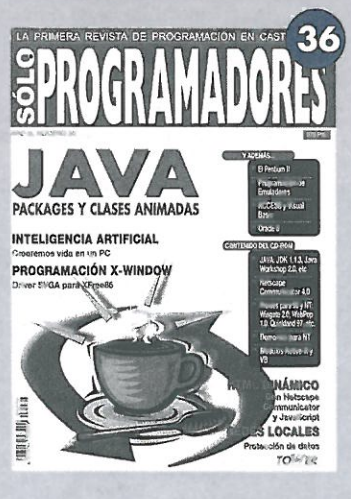
25
LA PRIMERA REVISTA DE PROGRAMACIÓN EN CAST
SÓLO PROGRAMADORES
Revista especializada para usuarios de PC
LINUX: PRESENTACIÓN DE TEXTO EN PANTALLA
WEB: DESCRIPCIÓN AVANZADA DE LA INTERFAZ CGI
TÉCNICAS DE COMPRESIÓN
ALGORITMO HUFFMAN

26
LA PRIMERA REVISTA DE PROGRAMACIÓN EN CAS
SÓLO PROGRAMADORES
Revista especializada para usuarios de PC
LINUX: EL GESTOR DE VENTANAS X-WINDOWS
WEB: LENGUAJE HTML
EUSKAL PARTY 96
EL FESTIVAL DE LA DEMOSCENE

28
LA PRIMERA REVISTA DE PROGRAMACIÓN EN CAS
SÓLO PROGRAMADORES
Revista especializada para usuarios de PC
ANALIZAMOS EL ESTADO DE LA INTELIGENCIA ARTIFICIAL
VISUAL C++ PROGRAMACIÓN DEL GDI
GRATIS CD-ROM CON:
ESPECIAL HERRAMIENTAS RAD
DESARROLLO RÁPIDO DE APLICACIONES CON OPTIMACRO







Listado 2: Código de la aplicación cliente echo.

```
unit Unit_cliente_echo;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ScktComp, StdCtrls, ExtCtrls;

type
  TForm1 = class(TForm)
    ClientSocket1: TClientSocket;
    EnviarEdit: TEdit;
    EnviarLabel: TLabel;
    EnviarButton: TButton;
    HostLabel: TLabel;
    HostEdit: TEdit;
    HostButton: TButton;
    Label1: TLabel;
    Memo1: TMemo;
    Label2: TLabel;
    Memo2: TMemo;
    Bevel1: TBevel;
    procedure HostButtonClick(Sender: TObject);
    procedure EnviarButtonClick(Sender: TObject);
    procedure ClientSocket1Connect(Sender: TObject;
      Socket: TCustomWinSocket);
    procedure ClientSocket1Read(Sender: TObject; Socket: TCustomWinSocket);
  end;

var
  Form1: TForm1;

implementation

{$R *.DFM}

procedure TForm1.HostButtonClick(Sender: TObject);
begin
  if (HostEdit.Text <> '') then begin
    if ClientSocket1.Active then ClientSocket1.Close;
    ClientSocket1.Host := HostEdit.Text;
    Memo1.Lines.Add ('Host remoto: ' + ClientSocket1.Host);
    if not ClientSocket1.Active then ClientSocket1.Open;
  end;
end;

procedure TForm1.EnviarButtonClick(Sender: TObject);
begin
  if EnviarEdit.Text <> '' then begin
    ClientSocket1.Socket.SendText (EnviarEdit.Text);
    Memo1.Lines.Add (EnviarEdit.Text);
  end;
end;

procedure TForm1.ClientSocket1Connect(Sender: TObject;
  Socket: TCustomWinSocket);
begin
  Memo1.Lines.Add ('Conectados con ' + ClientSocket1.Host);
end;

procedure TForm1.ClientSocket1Read(Sender: TObject;
  Socket: TCustomWinSocket);
begin
  Memo2.Lines.Add (Socket.ReceiveText);
end;

end.
```

información acerca de todas las conexiones activas con *sockets* clientes que el *socket* servidor ha aceptado. Las propiedades más destacadas de este objeto son:

- *SocketHandle*: Es el valor entero que se utiliza como descriptor del *socket* de una conexión.
- *Handle*: Es el *handle* de la ventana que recibe los mensajes de la conexión establecida con el *socket*.
- *Connections*: Es un array de objetos *TServerClientWinSocket,* cada uno de los cuales contiene cada una de las conexiones activas con *sockets* clientes. A su vez cada objeto *TServerClientWinSocket* de este array tiene propiedades que almacenan las direcciones y los números de puerto que utilizan el *socket* cliente y el *socket* servidor de cada conexión.

Una vez abierta una conexión, se puede obtener información de la misma a través del objeto TServerWinSocket que se crea con la conexión

- *AsyncStyles*: Propiedad que determina el tipo de eventos que se permiten recibir. Los eventos se recibirán en forma de mensaje de Windows en la propiedad *Handle*. Esta propiedad encapsula los eventos **FD_CONNECT**, **FD_READ**, etc., de la interfaz *Winsock*, y puede tomar como valor un conjunto de los siguientes: *asRead* (el *socket* recibe las notificaciones que informen que la conexión está preparada para leer datos), *asWrite* (conexión preparada para escribir), *asAccept* (un *socket*

solicita una conexión), *asConnect* (conexión con otro *socket* abierta), *asClose* (conexión con otro *socket* terminada).

SERVIDOR ECHO

De la misma manera que el mes pasado, donde desarrollamos un cliente *echo* para probar el correcto funcionamiento del *socket* cliente allí creado, vamos a crear ahora un sencillo servidor del protocolo **echo** para familiarizarnos con el componente *TServerSocket*. Este servicio podremos añadirlo posteriormente a nuestro servidor web como servicio añadido.

En la Figura 3 se muestra una pantalla con la aplicación servidora y la aplicación cliente del protocolo *echo* realizadas con los componentes de la paleta *Internet* dentro de Delphi. Para el servidor (que es el que nos interesa), se ha utilizado el componente *TServerSocket* y para el cliente, *TClientSocket*. El código de la aplicación servidora se muestra en el Listado 1, mientras que el del cliente se muestra en el Listado 2.'

Si queremos que haya posibilidad de enviar datos deberemos utilizar el comando Post

Lo primero que hemos hecho es colocar sobre un formulario de Delphi un componente *TServerSocket*. Después hemos añadido un componente *Label* y un *Memo* (este último recogerá información de las conexiones). Las propiedades de *TServerSocket* que hemos variado han sido las siguientes: *Service* (con el valor: **echo**), *Port* (con el valor 7) y *Active* (con el valor **True**), tal y como se muestra en la Figura 4. Con estos valores asignados a estas tres propiedades, el componente se va a poner a la escucha de peticiones en el puerto TCP número 7 nada más ejecutar la aplicación.

Ahora sólo falta añadir el código de respuesta a las peticiones de los clientes. Como el servidor *echo* se tiene que limitar a reenviar al cliente el texto que éste le envió, deberá recoger los datos del cliente y enviárselos de vuelta. Esto lo hará en el evento *OnClientRead* que informa que hay datos disponibles para ser leídos. En este evento, el segundo parámetro *Socket* es de tipo *TCustomWinSocket* y lleva toda la información sobre la petición del cliente. En concreto, uno de sus métodos es *ReceiveText* (recibir texto), que lee una cadena desde la conexión abierta y otro *SendText* (enviar texto) que envía una cadena sobre dicha conexión.

En los próximos artículos de esta sección crearemos un nuevo servidor HTTP

Así pues el manejador de este evento es sencillo:

```
procedure
      TForm1.ServerSocket1ClientRead(Sender:
      TObject;
  Socket: TCustomWinSocket);
var Recibido: string;
begin
  Recibido := Socket.ReceiveText;
  Memo1.Lines.Add ('Texto recibido: ' +
      Recibido);
  Socket.SendText (Recibido);
end;
```

Con esto tenemos el servidor *echo* terminado. Se le han añadido los manejadores de los eventos: *OnClientConnect* y *OnAccept*, para que muestre información del cliente y del estado de la conexión cuando ésta se establezca con el servidor. Se puede ver el resultado en el Listado 1.

CONCLUSIÓN

Hasta aquí la introducción a nuestro futuro servidor web. Hemos desentrañado algunos de los aspectos del protocolo *HTTP* y hemos creado un servidor del protocolo *echo* que nos ha servido para familiarizarnos con el componente *TserverSocket*, pieza fundamental para el desarrollo del servidor web. En el próximo artículo y con lo aprendido en éste, crearemos el servidor *HTTP* de acuerdo con la versión 1.0 del protocolo. A este servidor habrá que añadirle soporte a la versión más moderna del protocolo, la 1.1.

El Servidor Web que crearemos ofrecerá en un principio el servicio básico de enviar al navegador que lo solicite una página HTML y los recursos en ella referenciados. Así pues su primera función será comprobar que los recursos solicitados (página HTML y sus objetos) existan y en ese caso enviárselos al cliente. Como ya comentamos antes, si el navegador no especifica ninguna página concreta HTML, se enviará una por defecto. Tendremos, por tanto, previsto también ese caso.

Además, si queremos que haya posibilidad de enviarle datos desde un cliente, deberemos dar soporte a comandos *POST* y crear nuestros propios programas CGI para tratar los datos recibidos del cliente. Hasta el próximo artículo.

AUTOR

Enrique de la Lastra (elastra@redestb.es)

Si alguien desea enviar algún comentario sobre este artículo (o anteriores), puede hacerlo enviando un e-mail a la dirección de la revista:

solop@towercom.es
o a la dirección del autor:
elastra@redestb.es

LINUX

Programación de aplicaciones X-Window con Motif

Arturo Álvaro

Nuestro objetivo fundamental en este artículo es la introducción al programador de C en el desarrollo de aplicaciones X-Window mediante el uso de las librerías Motif. Para el desarrollo de las mismas, necesitaremos conocimientos básicos sobre programación orientada a objetos, en nuestro caso orientada al evento.

MODELO DE PROGRAMACIÓN X

Este modelo de programación se basa en un conjunto de herramientas que corresponden a librerías de tres niveles: *Xlib*, *Xt* y *Motif*. Estas librerías se encuentran vinculadas con un lenguaje anfitrión, en nuestro caso el lenguaje C.

El nivel más bajo está constituido por la librería *Xlib* que es una biblioteca de unas 300 subrutinas que gestionan el *display*, las ventanas y los dispositivos de entrada.

Sobre *Xlib* se construyó la librería *Xt Intrinsics* que tiene como función el uso de rutinas para manejar los objetos creados. Para el uso de este nivel se definen los siguientes ficheros de encabezamiento:

```
# include <X11/Intrinsic.h>
# include <X11/StringDefs.h>
```

Estos ficheros incluyen las funciones del nivel inferior *Xlib*.

El segundo objetivo de *Xt* consiste en inicializar la interfaz *Intrinsic* creando un contexto de aplicación y abriendo la conexión con el servidor X. También podemos hacer uso de las funciones de creación de objetos o *widgets* genéricos. Los *widgets* son objetos visibles en forma de ventana que suministran recursos comunes para sus descendientes, como por ejemplo un botón, un menú o un cuadro de diálogo.

Por último, utilizando la librería *Xt* se construyen diversas librerías de componentes gráficos que permiten la programación de interfaces de usuario como el caso de *Motif*.

Principalmente, utilizaremos las funciones de las librerías *Motif*, pero nos apoyaremos en algunas funciones de *Xt* que nos facilitarán la codificación de determinadas aplicaciones. Por lo tanto, antes de empezar con el nivel *Motif*, trataremos el modelo de programción *Xt*.

ESTRUCTURA DE UN PROGRAMA XT

Así como *Xlib* ofrece una programación de bajo nivel, *Xt* da un paso más y establece un modelo en el que se oculta todo el proceso de recepción, búsqueda y entrega de eventos. Para llevar a cabo este fin, el programador codifica subrutinas que tratan eventos específicos que ocurren sobre un determinado *widget*. Pero la ventaja principal de *Xt* sobre *Xlib*

reside en que la estructura del programa queda muy simplificada. Esta estructura es la siguiente:

1). *Ficheros de cabecera*: debemos incluir los ficheros necesarios para el uso del nivel *Xt,* indicados en el punto anterior. Además cuando vayamos a utilizar el nivel *Motif* debemos incluir el siguiente fichero:

```
# include <Xm/Xm.h>
```

Siempre que queramos utilizar una clase de *widgets* debemos incluir el fichero de cabecera correspondiente. Por ejemplo, la clase **XmPushButton** de *Motif* requiere el fichero <Xm/XmPushB.h> para utilizar un botón en nuestra nueva aplicación.

2) *Inicializar Intrinsics*: esta tarea se realiza mediante la rutina **XtInitialize** que lleva a cabo los siguientes pasos:

- Inicializa las variables internas de *Intrinsics*.
- Crea el contexto de aplicación, para la ejecución en paralelo de nuestra aplicación en el sistema multitarea de Unix.
- Abre conexión con el servidor X.
- Crea el primer *widget* de la aplicación, que según la estructura jerárquica que proporciona *Xt*, será el padre de todos los *widgets* que creemos.

Los parámetros que puede recibir esta rutina son los siguientes:

```
Widget XtInitialize{
  char*                Nombre de la aplicación,
  char*                Clases de la aplicación,
  XrmOptionDescList    Opciones,
  int                  Número de opciones,
  int*                 argc,
  char**               argv};
```

El argumento *argc* es el número de parámetros que toma la aplicación desde la línea de comandos. El argumento *argv* es una array de *argc* posiciones que contiene los parámetros incluidos en la línea de comandos.

3) *Creación de los widgets*: sobre el *widget* raíz que crea **XtIntrinsics** serán creados los demás *widgets* de nuestra aplicación. La rutina de creación de un *widget* es la siguiente:

```
Widget XtCreateWidget{
  char*                Nombre,
  WidgetClass          Clase,
  Widget               Padre,
  Arglist              Lista de Argumentos,
  int                  Número de Argumentos};
```

4) *Definir y registrar los callbacks*: se trata de llamadas a rutinas producidas por un evento efectuado sobre un *widget*. La rutina que añade un *callback* al manejador de eventos es la siguiente:

```
XtAddcallback{
  Widget               widget,
  String               NombreCallback,
  XtCallbackProc       Rutina callback,
  XtPointer            Datos de la aplicación};
```

5) *Dar vida a los widgets*: un widget es visible en la pantalla cuando él y sus antecesores se encuentran manejados, realizados y mapeados. Los *widgets* padres actúan como responsables del manejo de sus hijos, asignándoles una posición en pantalla. La rutina para manejar un *widget* es la siguiente:

```
void XtManageChild(
    Widget          widget);
```

Se dice que un *widget* está realizado cuando tiene asociado una ventana de X-Window. La rutina de *Xt* para el manejo de un *widget* es:

```
void XtRealizeWidget(
    Widget          widget);
```

Se dice que un widget está mapeado cuando su ventana aparece visible en pantalla. La rutina de *Xt que* necesitaremos para efectuar este mapeado es la siguiente:

```
void XtMapWidget(
    Widget          widget);
```

Sin embargo, es posible que un *widget* quede creado y manejado con el uso de la siguiente rutina:

```
XtCreateManagedWidget(
  char*                Nombre del Widget,
  WidgetClass          Clase,
  Widget               WidgetPadre,
  ArgList              Lista de Argumentos,
  Int                  Longitud de la lista);
```

6) *Entrar en el bucle de manejo de eventos*: consiste en activar un bucle infinito de eventos recibidos del servidor y que son enviados al *callback* correspondiente. La rutina que inicia el bucle de manejo de eventos es la siguiente:

```
void XtMainLoop();
```

Como podemos ver la estructura del programa *Xt* está dividida en seis partes que se corresponden con el orden real de la codificación. A continuación podemos apreciar esta estructura en el siguiente ejemplo que nos sirve para llegar a una mejor comprensión de la programación en el nivel *Xt*.

Argc es el número de parámetros que toma la aplicación desde la línea de comandos

En este primer ejemplo se utilizan algunos conceptos, que aunque aún no hemos visto todavía, sirven para clarificar la estructura del programa en Xt. Esta primera aplicación creará dos botones en una ventana con un separador entre ellos. El primer botón denominado **Xload** permite ejecutar en un terminal virtual la instrucción **ps** del sistema Unix, mientras que el segundo botón denominado **Salir**, termina la ejecución de la aplicación

```
/*   1) Ficheros de cabecera   */
# include <X11/Intrinsic.h>
# include <X11/StringDefs.h>
# include <Xm/Xm.h>
# include <Xm/PushB.h>
# include <Xm/RowColumn.h>
# include <Xm/Separator.h>

void carga(Widget widget,Widget
        DatosAplicacion,XtPointer DatosWidget);
void salir(Widget widget,Widget
        DatosAplicacion,XtPointer DatosWidget);

/* declaraciones de variables globales*/
int n;
Arg args[10];
XmString cadena;

main (int argc,char* argv[])
{
Widget toplevel,base,boton1,boton2,separador;

/*2) Inicializacion del nivel Xt*/

toplevel=XtInitialize("ejemplo1","Ejemplo1",NUL
        L,0,&argc,argv);

/*3) Creacion de la jerarquia de widgets y su mane-
        jo*/

base=XtCreateManagedWidget("Base",
        xmRowColumnWidgetClass,toplevel,args,n);
boton1=XtCreateManagedWidget ("Xload",
        xmPushButtonWidgetClass,base,args,n);
separador=XtCreateManagedWidget("sp",xmSepar
        atorWidgetClass,base,args,n);
boton2=XtCreateManagedWidget("Salir",
        xmPushButtonWidgetClass,base,args,n);

/*4) Callbacks*/

XtAddCallback(boton1,XmNactivateCallback,carga,
        NULL);
XtAddCallback(boton2,XmNactivateCallback,salir,N
        ULL);

/*5) Realizacion de la jerarquia*/

XtRealizeWidget(toplevel);

/*6) Bucle de espera de eventos*/

XtMainLoop();

}
```

```
/*Callback que muestra los procesos del sistema*/

void carga(Widget widget,Widget
        DatosAplicacion,XtPointer DatosWidget)
{
system("xterm -e scr &");
}

/*Callback que permite salir de la aplicacion*/

void salir(Widget widget,Widget
        DatosAplicacion,XtPointer DatosWidget)
{
exit(0);
}
```

No debe preocuparnos entender todas las instrucciones del programa, ya que la finalidad del ejemplo es que comprendamos la importancia de los seis puntos de la estructura del programa *Xt*.

ASIGNACIÓN DE VALORES A LOS RECURSOS DE LOS WIDGETS

Ya hemos hablado antes de la importancia de la estructura jerárquica en este modelo de programación, y es ahí donde quedan definidos los valores por defecto que los tomarán *widgets*. Todos los *widget* de nuestra aplicación toman por defecto para todos sus recursos los valores especificados en la clase y sus superclases. Cuando hablemos de cada uno de los *widgets* que podemos crear mostraremos esa relación jerárquica que nos aclarará un poco más esta idea propia de la programación orientada a objetos.

Por el momento nos basta con saber que los valores que toma un *widget* de su clase pueden ser modificados por el programador. La rutina que nos permite esta función es la que os indicamos a continuación.

```
XtSetArg(
        Arg         Argumento,
        char*       Nombre del recurso,
        XtArgVal    valor que se asigna);
```

Como se muestra en el siguiente ejemplo podemos establecer los valores de los recursos de un *widget* con la siguiente codificación:

```
/* sección de variables globales */

Arg   Argumentos[10];
int   n;

...

/* creación del widget */

n=0;
XtSetArg(Argumentos[n],XmNx,175);n++;
XtSetArg(Argumentos[n],XmNy,110);n++;
XtSetArg(Argumentos[n],XmNwidth,50);n++;
XtSetArg(Argumentos[n],XmNheight,30);n++;
MiWidget=XtCreateWidget("ejemplo",xmPushBut
        tonWidgetClass,WidgetPadre,
        Argumentos,n);
...
```

Así, mediante la asignación entre pares recurso–valor, establecemos los valores en la creación de los *widget*. En el ejemplo anterior hemos creado un botón que se situará con respecto a su padre en el punto geométrico (175,110), midiendo en *pixels*, y que tendrá una anchura de 50 y una altura de 30. Más adelante veremos por clases los recursos que podemos utilizar en la creación de nuestros *widgets*.

Xlib ofrece programación de bajo nivel

En esta primera aproximación a la programación de aplicaciones X-Window, hemos dado unos pasos introductorios en el conocimiento de la programación de una aplicación *Xt*. En próximas entregas examinaremos detenidamente todos los tipos de *widgets* que podemos incluir en nuestras aplicaciones llegando por fin al desarrollo de aplicaciones orientadas a cualquier nivel en el que trabajemos.

Servlets (II): La cara oculta de la web

Juan Carlos Sastre Iglesias (jcsi@geocities.com)

En esta segunda entrega del curso se van a seguir todos los pasos necesarios para hacer funcionar sin problemas un servlet, y se va a realizar un primer análisis de la estructura básica.

Para empezar con este tema utilizamos un ejemplo sencillo que nos va a servir para entender la forma básica de trabajar. El ejemplo va a consistir en una página HTML en la que aparecerá un campo de entrada con los correspondientes botones **Submit** y **Reset** típicos de cualquier formulario de entrada de datos.

Al pulsar sobre el primero de ellos se producirá la ejecución del servlet en el servidor, que procesará la información recibida, y mostrará como resultado una página HTML con el texto introducido con anterioridad.

CÓDIGO HTML

En el listado 1 se muestra el código HTML necesario para implementar el ejemplo que nos hemos propuesto.

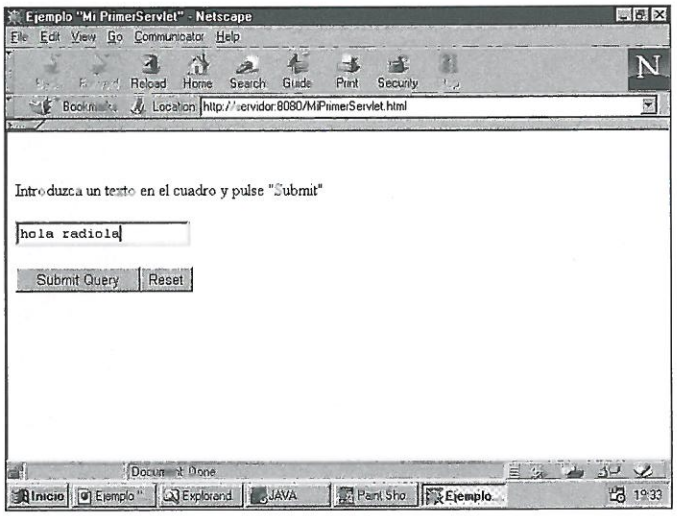
Dentro del mismo destacan dos partes: en primer lugar la cabecera del formulario, que indica la acción a desencadenar en el servidor al pulsar el botón **Submit**, y en segundo lugar la parte en que se define el propio formulario. El fichero que contiene la página en formato de documento HTML se llamará **MiPrimerServlet.html**, y lo guardaremos en el subdirectorio **/public_html** dentro del directorio donde hayamos instalado el *JWS*.

CÓDIGO DEL SERVLET

Para este primer ejemplo, el código necesario va a ser mínimo, como se puede observar en el Listado 2, lo que significa que es una prueba de la enorme potencia y simplicidad derivada del uso de servlets. En el programa destaca principalmente la utilización de los paquetes **javax.servlet.*** y **javax.servlet.http.***, y la cláusula extends que define el programa de tipo HttpServlet. El fichero que contiene el servlet lo llamaremos MiPrimerServlet.java y tras compilarlo con el compilador del JDK (javac), almacenaremos el fichero .class resultante en el directorio /servlets dentro del directorio donde hayamos instalado el JWS.

CONFIGURACIÓN DEL SERVLET EN EL JWS

Para que el servlet funcione sin problemas en el JWS es necesario darlo de alta. En primer lugar hay que activar el JWS ejecutando el programa *httpd.exe* que se encuentra en el directorio */bin*. Una vez activado, con el navegador que deseemos accederemos a la parte de administración. Para acceder a la página principal habría que introducir



En esta figura se muestra el campo de entrada con de la página HTML, con los botones Submit y Reset pertenecientes a un formulario de entrada de datos.

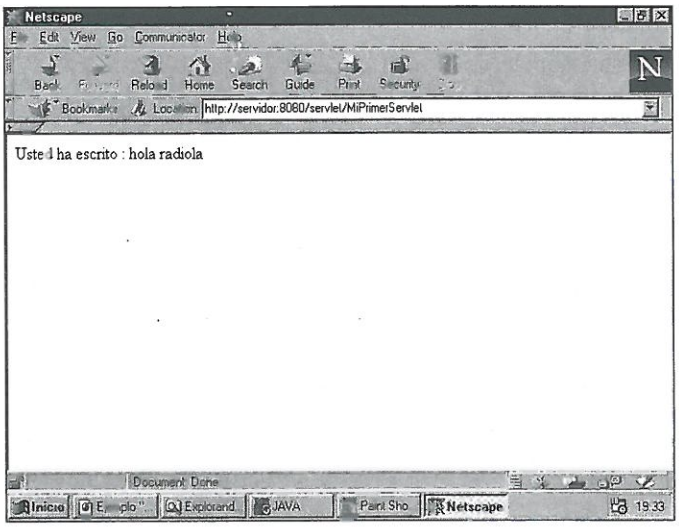
Listado 1. Página HTML

```
<html>
<head>
<title>Ejemplo "Mi Primer Servlet"</title>
</head>
<body>
<form action=http://servidor:8080/servlet/MiPrimerServlet method=POST>
<BR>
<BR>Introduzca un texto en el cuadro y pulse "Submit"<BR>
<BR>
<input type=text name=TEXTO>
<BR>
<BR><input type=submit><input type=reset></form>
</body>
</html>
```

una dirección como http://[nombre_servidor_DNS]:8080/, y desde aquí podremos acceder al applet de Administración. Tras introducir nombre y contraseña del *Administrador* (por defecto admin/admin), aparecerá una página donde se podrá escoger el servicio que deseamos configurar.

El servlet más simple posible es aquél que sólo implementa el método service

En nuestro caso seleccionaremos **Web Service**, tras lo cual aparecerá una ventana donde deberemos pulsar sobre el botón **Servlets**. El programa mostrará una estructura de tipo árbol, donde seleccionaremos **Add**. A continuación introduciremos los datos, tal y como indica la figura 4 y pulsaremos de nuevo sobre **Add**. Además será necesario introducir una serie de datos adicionales, tras lo cual pincharemos sobre **Save**. Tras efectuar los pasos indicados, podremos salir y realizar la ejecución del ejemplo.



Aquí se muestra una página HTML, con el texto introducido con anterioridad.

EJECUCIÓN DEL SERVLET DE EJEMPLO

Para ejecutar y comprobar el correcto funcionamiento de nuestro servlet, solamente tendríamos que introducir en nuestro navegador una dirección como http://[nombre_servidor_DNS]:8080/MiPrimerServlet.html.

El método service

El servlet más simple posible es aquél que sólo implementa el método **service**, el cual se encarga de procesar la petición del cliente. Utiliza dos parámetros, uno de petición y otro de respuesta. En el caso particular del protocolo *http* se utilizan instancias de las clases (o interfaces) **HttpServletRequest** y **HttpServletResponse** respectivamen-

te. Estos parámetros encapsulan los datos que son enviados tanto desde el cliente, como desde el servidor. De hecho, normalmente se utilizan **input streams** para recuperar datos desde el cliente, y **output streams** para responder. En nuestro caso utilizamos **PrintStream** para responder al cliente.

Utilizamos PrintStream para responder al cliente

Para recuperar los datos que manda el cliente, utilizamos el método **getParameter**, que es heredado por **HttpServletRequest** de la interfaz *ServletRequest*. Finalmente, utilizando el stream de salida se envía la respuesta al cliente. Como puede verse, el programa es muy sencillo, pero la funcionalidad muy alta.

Ejercicio

Cómo ejercicio para el lector se propone la implementación de un servlet similar al anterior. Deberá aparecer un formulario HTML, con dos campos de entrada. Un primer campo para introducir el nombre, y otro para introducir la dirección e-mail. El servlet deberá responder con los datos introducidos, incluyendo además la dirección IP desde donde está conectado el cliente. Para obtener esta dirección IP se debe utilizar el método **getRemoteAddr** de la interfaz *HttpServletRequest* (en la tabla 1 se encuentra toda la información necesaria sobre este método).

UNA DEFINICIÓN MÁS FORMAL

Podríamos definir un Servlet como un programa JAVA que se ejecuta en el marco de un servicio de red, como por

Tabla 1. El método getRemoteAddr

getRemoteAddr
public abstract String getRemoteAddr()
Devuelve la dirección IP del cliente que envía la petición. Es similar a la variable CGI REMOTE_ADDR.

Cuadro 1. Un poco de teoría

Una definición más formal

Podríamos definir un Servlet como un programa JAVA que se ejecuta en el marco de un servicio de red, como por ejemplo un servidor web, y que recibe y responde a las peticiones de un cliente.

El Ciclo de Vida

Los servlets al igual que los applets poseen un Ciclo de Vida perfectamente definido controlado por el entorno (una vez más se podría poner como ejemplo un servidor web). La interfaz que se utiliza para implementarlos define una serie de métodos para cada una de sus etapas, y como ya se ha comentado, es el entorno el que se encarga de invocarlos. El orden seguido es el siguiente:

1) Cuando el servidor carga el servlet (solo lo carga ante la primera petición, después ya no hace falta), invoca el método **init**. Todas las peticiones que lleguen antes de que este método finalice su ejecución habrán de esperar para ser atendidas.

2) Una vez finalizada la ejecución de dicho método, todas las peticiones son atendidas por el método **service**. Normalmente el servidor puede manejar varias peticiones simultáneamente y asigna un hilo propio (*thread*) a cada petición.

3) Cuando el servidor quiere descargar el servlet invoca el método **destroy**.

En un próximo capítulo se estudiarán con mayor detalle estos métodos.

El API de los servlets

El API utilizado para la programación de Servlets incluye dos paquetes:

El paquete **javax.servlet**, que es la base, y que define la interfaz *Servlet* y su implementación **GenericServlet**. Es esta la interfaz que define los métodos del Ciclo de Vida. También destacan las interfaces *ServletRequest* y *ServletResponse*, que definen la mecánica de las peticiones y las respuestas.

El paquete **javax.servlet.http**, que es una especialización del paquete anterior, se utiliza con el protocolo *http*. Este paquete define la clase **HttpServlet** que es una extensión de **GenericServlet**. Asimismo incluye las interfaces *HttpServletRequest* y *HttpServletResponse*, que son extensiones de *ServletRequest* y *ServletResponse* respectivamente.

En un próximo capítulo también se estudiarán en profundidad estos dos paquetes, cuya aportación para el trabajo con servlets es fundamental.

ejemplo un servidor web, y que recibe y responde a las peticiones de un cliente.

El Ciclo de Vida

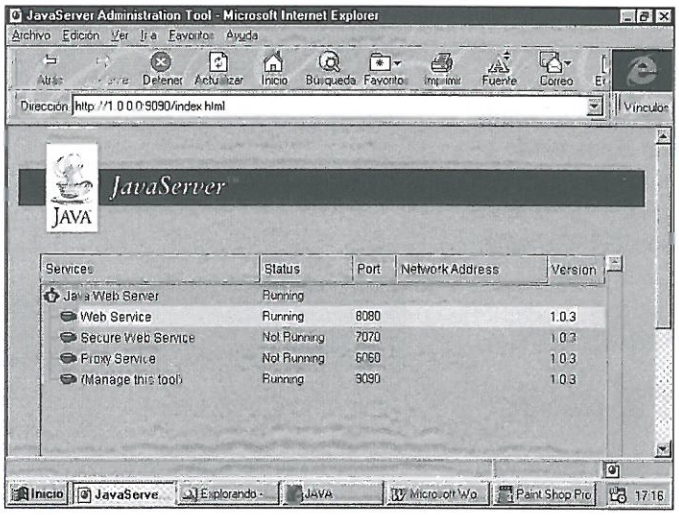
Los servlets al igual que los applets poseen un Ciclo de Vida perfectamente definido controlado por el entorno (una vez más se podría poner como ejemplo un servidor web). La interfaz que se utiliza para implementarlos define una serie de métodos para cada una de sus etapas, y como ya se ha comentado, es el entorno el que se encarga de invocarlos. El orden seguido es el siguiente:

1). Cuando el servidor carga el servlet (solo lo carga ante la primera petición, después ya no hace falta), invoca el método **init**. Todas las peticiones que lleguen antes de que este método finalice su ejecución habrán de esperar para ser atendidas.

2) Una vez finalizada la ejecución de dicho método, todas las peticiones son atendidas por el método **service**. Normalmente el servidor puede manejar varias peticiones simultáneamente y asigna un hilo propio (*thread*) a cada petición.

3) Cuando el servidor quiere descargar el servlet invoca el método **destroy**.

En un próximo capítulo se estudiarán con mayor detalle estos métodos.



Como se hace referencia en el texto citado, esta figura muestra una página que permite escoger el servicio que deseemos configurar.

El API de los servlets

El API utilizado para la programación de Servlets incluye dos paquetes:

El paquete **javax.servlet**, que es la base, y que define la interfaz *Servlet* y su implementación **GenericServlet**. Es esta la interfaz que define los métodos del Ciclo de Vida. También destacan las interfaces *ServletRequest* y *ServletResponse*, que definen la mecánica de las peticiones y las respuestas.

Los servlets poseen un Ciclo de Vida perfectamente definido por el entorno

El paquete **javax.servlet.http**, que es una especialización del paquete anterior, se utiliza con el protocolo *http*. Este paquete define la clase **HttpServlet** que es una extensión de **GenericServlet**. Asimismo incluye las interfaces *HttpServletRequest* y *HttpServletResponse*, que son extensiones de *ServletRequest* y *ServletResponse* respectivamente.

En un próximo capítulo también se estudiarán en profundidad estos dos paquetes, cuya aportación para el trabajo con servlets es fundamental.

Listado 2. Servlet Java

```
//
//  MiPrimerServlet.java
//

import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;

public class MiPrimerServlet extends HttpServlet
{

  public void service(HttpServletRequest req, HttpServletResponse res)
         throws ServletException, IOException
  {
   PrintStream out = new PrintStream(res.getOutputStream());
   res.setContentType("text/html");
   String TEXTO    = req.getParameter("TEXTO");
   out.println("<p>Usted ha escrito : "+TEXTO+"</p>");
  }

}
```

MicroFocus NetExpress

Javier Amado (jamado@towercom.es)

¿QUÉ PROGRAMADOR NO UTILIZÓ EN SUS INICIOS COBOL?

Desde los principios del lenguaje COBOL, hasta el día de hoy éste ha sufrido muchas modificaciones y son muchos los dialectos que existen actualmente. Pero a pesar de lo que algunos programadores creen el COBOL no está muerto.

Un 70% de la información del negocio mundial reside en mainframes y existen 150.000 líneas de código en COBOL. Las empresas que disponen de este tipo de sistemas que de un modo efectivo se puedan reutilizar sus mainframes heredados.

PRIMER CONTACTO

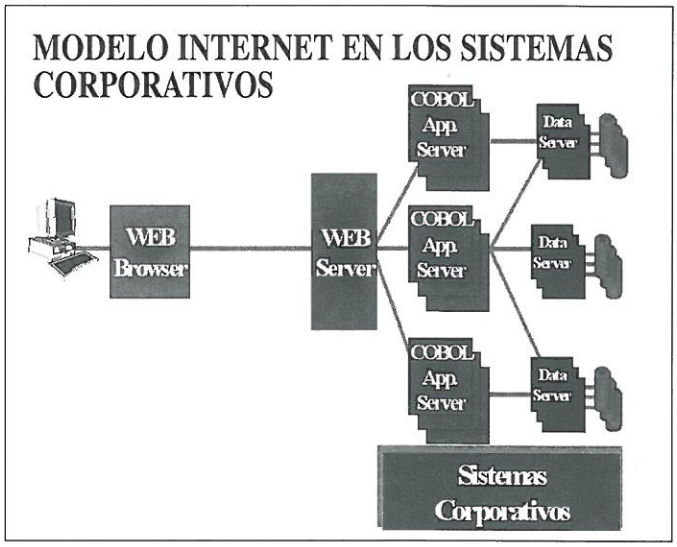
NetExpress es la primera solución que permite a las empresas reutilizar el código COBOL existente para proporcionar acceso a datos corporativos críticos, utilizando por ejemplo las últimas tecnologías WEB. Es decir, programas creados en código COBOL, pueden actualizarse y cambiar la sobria ventana en modo texto por un interfaz bajo Windows. La aplicación origen es ejecutada en segundo plano mientras todo el contacto que existe entre ésta y el usuario se realiza mediante un interfaz bajo Windows. Con lo que ello conlleva, una mayor facilidad en su utilización y un interfaz más agradable. NetExpress es una aplicación 32 bits que trabaja bajo entornos Windows (95 o NT 4.0), necesitando 16 Mbytes de RAM y 70 Mbytes de disco duro, como mínimo.

FUNCIONAMIENTO BAJO DISTINTOS ENTORNOS

Una de las ventajas más destacables que ofrece esta aplicación es que permite la posibilidad de generar código portable entre numerosos sistemas operativos y plataformas hardware, incluyendo sistemas basados en Windows NT y en UNIX. De este modo podemos utilizar el código ya existente, como por ejemplo, aplicaciones que se utilizan desde hace tiempo por un grupo de clientes y enfrentarlo a un servidor UNIX, e incluso ejecutarlo desde el propio servidor

NetExpress es un entorno de desarrollo para crear, mantener y desplegar tanto aplicaciones intranet/Internet como aplicaciones gráficas distribuidas, permitiendo reutilizar código antiguo y actualizarlo para su ejecución tanto en Internet como en intranets.

UNIX, mediante un cliente basado en un sistema operativo Windows.



Enfrentado al modelo Cliente / Servidor algunas empresas ven como futuro en las redes coporativas el modelo Internet, que poco a poco aunque a ritmo lento se está implantando en algunas empresas.

"EL COMPILADOR" EL NÚCLEO DE TODO EL SISTEMA DE DESARROLLO

En este tipo de aplicaciones el compilador es el núcleo de la mayoría de los procesos. Gracias a él obtendremos un resultado más optimizado dependiendo de la máquina y el sistema operativo. Por ello hemos creído importante resaltar las características que este proporciona:

- El compilador proporciona toda la funcionalidad exigida a un entorno de programación orientado a objetos incluyendo extensas librerías de clases y sus correspondientes browsers.
- Permite generar código nativo de gran rendimiento que puede ser optimizado para obtener las máximas prestaciones de plataformas hardware específicas.
- Escalabilidad de las aplicaciones, desde los PCs independientes hasta los sistemas UNIX de tipo multiprocesador.

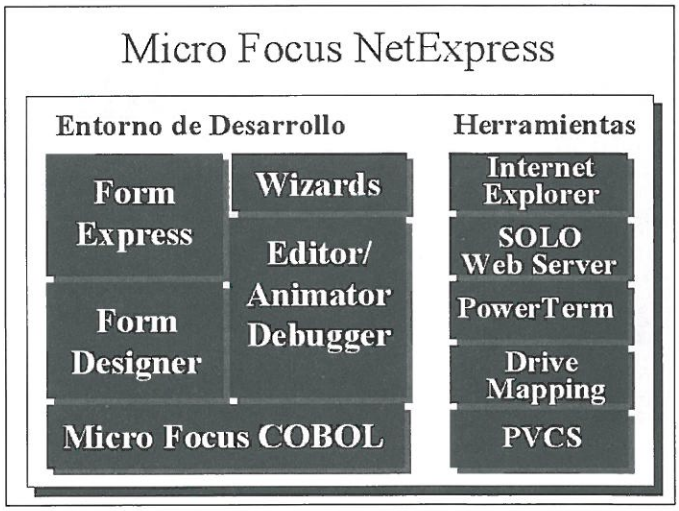
Tabla 1.Características principales como Entorno de desarrollo

- Desarrollo por Proyectos
- Asistentes
- Editor/Animador con plenas funcionalidades
- Vistas múltiples (Multiple Document Interface)
- Control de generación de ejecutables (DLL o EXE)
- Sistema de Control de Versiones
- Micro Focus COBOL (Dialectos/Compatibilidades)
  - ANSI 85, 74
  - OS/VS, VSII COBOL 370 DOS/VS, SAA, VSII, COBOL/2
  - RM COBOL, MS COBOL, DG/I COBOL
  - FIPS Standard, X/Open (XPG4)
- Soporta Multi-threading
- Orientación a objetos
- Genera ejecutables
- Comunicaciones:
  - Soporta los principales interfaces de Web Server
  - CGI
  - ISAPI (Microsoft)
  - NSAPI (Netscape)
  - Soporta Microsoft Transaction Server
  - OLE: Facilidad de generación de objetos OLE en COBOL
  - Interface a los ORB compatibles CORBA

INTERNET-COBOL, COBOL-INTERNET

Muchos programadores utilizan como medio Internet parar realizar consultas en bases de datos, por medio de páginas WEB de su servidor. Por regla general estas consultas son realizadas mediante búsquedas en lenguaje SQL. Ahora esto ya no es necesario, ya que NetExpress dispone de la posibilidad de realizar formularios CGIs para consultas en bases de datos gestionadas mediante COBOL por medio de Internet o Intranets. Mediante el entorno de desarrollo gráfico que proporciona esta aplicación será posible realizar formularios para mantener o consultar los datos de un programa escrito estrictamente en COBOL. Únicamente pulsando un botón se analiza el código y se obtiene el formulario. El resultado es un tanto escueto ya que sólo son presentados los nombres de los campo y estos. Para disponer de una página más atractiva será necesario editar esta con otro programa para añadir complementos, fondo y

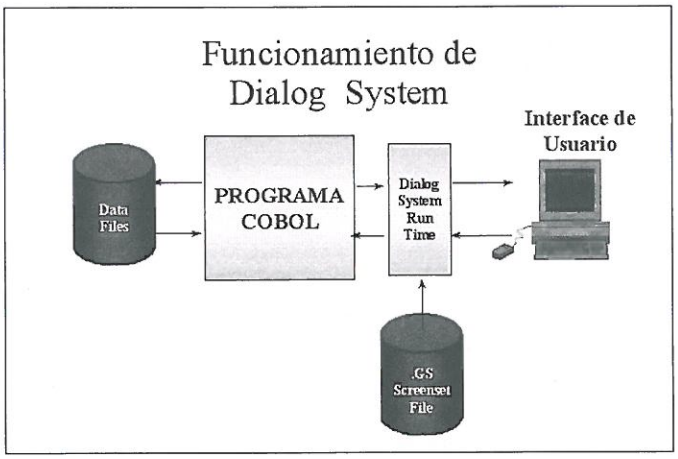
todo aquello que sea pertinente. En el campo de las comunicaciones se han implementado soporte para los principales interfaces, como: CGIs, ISAPI (Microsoft), NSAPI(Netscape), Microsoft Transaction Server, OLE que facilita la generación de objetos OLE en COBOL.



En este esquema se puede comprobar la estructuración de NetExpress.

COMPATIBILIDAD COBOL

Tal y como se especificaba al principio existen muchos dialectos del COBOL. Desde sus inicios este lenguaje ha sufrido muchas modificaciones para poder adaptarse a los nuevos sistemas Cliente-Servidor o a las comunicaciones. Por ello cuando se desee obtener un resultado óptimo en las actualizaciones, es complicado asegurar que NetExpress soportará ese dialecto en concreto. Según se especifica en la tabla 1 estos son los dialectos soportados por la aplicación ante los cuales no existe ningún contratiempo. Como se puede apreciar en la tabla no se encuentra COBOL 400, COBOL específico para sistemas AS-400. Pero esto no significa que no se puedan llevar a cabo actualizaciones en este tipo de archivos. No existe ningún tipo de problema en cuanto al código se refiere, pero en el caso de las implementaciones del interfaz, no puede ser generada de modo automático por el programa ya que este dialecto dispone de una serie de características en lo que a las pantallas se refiere. En este caso se tendrán que crear las pantallas manualmente mediante el editor de entornos gráficos. Pero esto no es un problema mayor, ya que se dispone de asistente para guiarnos a través de todo el desarrollo de éste.



Herramienta para crear interfaces gráficos (GUI) que ofrecen facilidad de uso por parte de los programadores COBOL y facilidad de conexión del GUI a través de las redes.

Tabla 2. Capacidades Avanzadas de Acceso a Datos

- OpenESQL y Asistente para OpenESQL
- Acceso a SGBDR desde SQL embebido via ODBC
- Acceso ESQL — No se requieren conocimientos sobre ODBC
- ODBC drivers incorporados al producto
- Oracle, Sybase, SQL Server, Access
- Genera el código ESQL
- Micro Focus File Handler Mejorado
- Admite ficheros de hasta 512GB

RESUMIENDO

NetExpress es un entorno de desarrollo integrado que tiene como fundamento la productividad del programador y disminuye los gastos en formación a los usuarios. También combina Object COBOL con un conjunto de herramientas que reduce el tiempo requerido para la puesta a punto de nuevas aplicaciones de última tecnología. Permite a las empresas reutilizar los datos y la lógica de la empresa, como las subrrutinas. Del mismo modo NetExpress incorpora herramientas visuales que permiten a los programadores crear directamente aplicaciones para la WEB, como: Form Designer o Form Express. Aunque los entornos informáticos distribuidos no están aún muy definidos se soportan todos los estándares que lideran el mercado. Por lo tanto se dispone de una opción de renovación de las antiguas aplicaciones, sobre todo para aquellas empresas que pensaban que sus sistemas basados en líneas programadas en COBOL eran muy lentas.

HERRAMIENTAS DE PROGRAMACIÓN

C++ Builder 3 A la tercera va la vencida

José Antonio Romero (jaromero@openbank.es)

C++ Builder 3 es la evolución natural de las herramientas de desarrollo en C++, al conjugar la rapidez de la programación visual con la potencia del lenguaje C++. Está pensado para resolver las necesidades de los desarrolladores de Sistemas (programación de aplicaciones en tiempo real, drivers, etc.) y de Aplicaciones.

Para quien haya desarrollado con Delphi 3 o con C++ Builder, su forma de trabajo no variará demasiado, ya que seguimos teniendo el mismo entorno de desarrollo, eso sí, con grandes mejoras. Para los que no lo conozcan, decir que consta de tres componentes principales (Figura 1):

C++ Builder 3 destaca porque reduce considerablemente el tiempo de desarrollo

1. *Paleta de Componentes*. Donde residen todos las piezas que vamos a utilizar para ensamblar nuestra aplicación. Tiene más que la versión anterior. Ver Tabla 1.
2. *Object Inspector*. Presenta las propiedades y métodos de los componentes.
3. *Editor de Código*. Ventana en la que se irá introduciendo el código correspondiente a cada función desarrollada.

Este compilador se nos va a presentar en diferentes versiones, para que escojamos según nuestras necesidades.

VERSIONES DE C++ BUILDER 3

- **Enterprise**. Desarrollo de aplicaciones distribuidas multi-tier con características tales como thin-client, equilibrio de carga, objetos distribuidos y alto volumen de transacciones. Soporta CORBA, DCE, TCP/IP y DCOM. Incluye las herramientas de la siguiente versión.
- **Cliente/Servidor**. En este tipo englobaríamos los sistemas de soporte a la toma de decisiones, servidores y clientes Internet y otros sistemas similares. De nuevo, incluye las herramientas de la versión profesional.
- **Profesional**. Nos permite crear casi cualquier aplicación Windows. Entre ellas incluiríamos las aplicaciones de consola, los drivers de los dispositivos, y otras aplicaciones no visuales. Entre las visuales destacaríamos las aplicaciones de bases de datos y de Internet. Esta versión incluye todas las características de la estándar.
- **Estándar**. Para desarrolladores no profesionales que buscan una forma productiva de aprender o mejorar sus conocimientos de C++.

- **C++ Builder/400.** Para las personas que quieran explotar la potencia del AS/400 por medio de sofisticadas aplicaciones Windows, Web o de base de datos, accediendo directamente al AS/400.

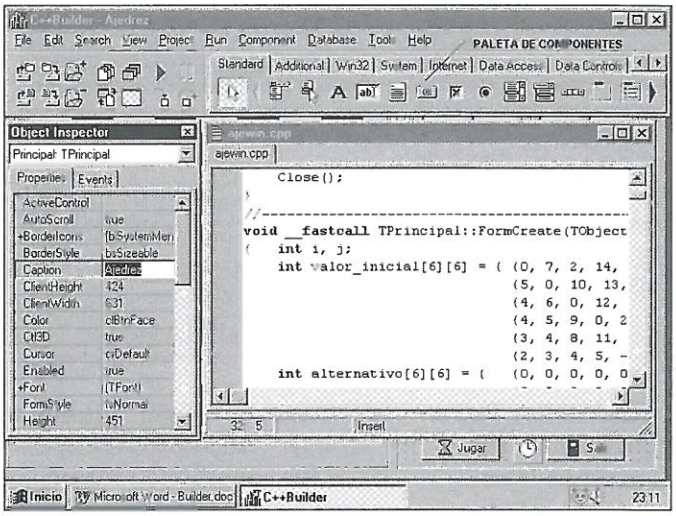
Ofrece compatibilidad con el código C y C++ existente, así como con las OWL y las MFCs

La versión sobre la que hemos escrito este artículo es la Professional. Sus requisitos mínimos pueden verse en la Tabla 2.

Durante el proceso de instalación será posible elegir entre cuatro posibilidades, con el fin de transferir al disco duro únicamente los componentes deseados (Tabla 3).

CARACTERÍSTICAS TÉCNICAS DEL PAQUETE

C++ Builder 3 destaca porque reduce considerablemente el tiempo de desarrollo, apoyándose en la gran facilidad que nos proporciona para generar interfaces de usuario, gracias a su Librería de Componentes Visuales (*VCL*). La utilización de este conjunto de componentes no implica una disminución del rendimiento del código C++ generado. Esta librería tiene más de 130 componentes perfectamente integrados entre sí, destacando por el hecho de estar compartida con Delphi, lo que conlleva un considerable aumento de la productividad.



Entorno de Desarrollo de C++ Builder 3.

Además, va a mantener la compatibilidad con el código C y C++ existente, así como con los marcos de desarrollo de las *OWL* (*Objects Windows Library*) y de las *MFC* (*Microsoft Foundation Class*). También tiene soporte para interfaces estándares como ActiveX, COM y CORBA.

Una Arquitectura de Componentes Orientada a Objetos

ENTORNO DE DESARROLLO VISUAL

Este entorno se basa en:

- **Una Arquitectura de Componentes Orientada a Objetos.** El factor clave en el desarrollo visual es la **reutilización** de código. Para obtener unos altos niveles de productividad, el programador de C++ debe ser capaz de incluir en sus aplicaciones una gran cantidad de códi-

Tabla 1.- Grupos de Componentes

Grupo	Descripción	Componentes
Standard	Componentes presentes en cualquier aplicación Windows	PopupMenu, ComboBox, Button,...
Additional	Más componentes Windows	BitBtn, SpeedButton,...
Win32	Controles Windows 95 y NT	TreeView, ListView,...
System	Controles OLE y multimedia	MediaPlayer, OLEContainer,...
Internet	Protocolos y componentes de acceso a Internet	HTML, NMPOP3, NMFTP,...
Data Access	Tratamiento de los Datos de una Base de Datos	DataSource, Table,...
Data Controls	Visualización de Datos	DBGrid, DBEdit,...
Qreports	Diseño de Informes	QuickRep, QRGroup,...
Dialogs	Diálogos de sistema ya generados	OpenDialog, SaveDialog,...
Win 3.1	Controles existentes en Windows 3.1	FileListBox, DirectoryListBox,...
Samples	Ejemplos	Calendar,...
ActiveX	Controles ActiveX ya generados.	Graph, F1Book,...

Tabla 2. Requisitos Mínimos del Sistema

Intel 486DX/100 MHz o Superior.
Microsoft Windows 95 o Windows NT (Service Pack 2).
24 MB de RAM (Recomendables 32 MB).
Como mínimo, 175 MB en disco duro.
Unidad de CD-ROM.
Monitor de resolución VGA o superior.
Ratón.

go previamente desarrollado. De esta forma se obtiene una gran ventaja a la hora de abordar un nuevo proyecto, puesto que parte del trabajo previo ya se encuentra realizado, y únicamente tendremos que centrarnos en las especificaciones del propio proyecto. Esto se consigue con la VCL, que divide sus componentes en cinco categorías:

1) Interfaces de Usuario.
2) Bases de Datos y soporte a la toma de decisiones.
3) Sistema.
4) Multimedia.
5) Internet y Web.

- **Desarrollo del Interface de Usuario por medio de Drag & Drop.** C++ Builder contiene herramientas de diseño visual para la integración de componentes y la introducción de código. Esto implica que el código se puede mantener a través de las herramientas visuales y de los editores de código tradicionales. Los cambios realizados en uno, serán recogidos por el otro gracias a la tecnología *Two-Way-Tools*.

La forma de trabajar consiste en crear un formulario y, sobre él, colocar los componentes requeridos. Estos componentes pueden mantener sus propiedades de forma individual a través del *Object Inspector*. Simultáneamente, se representan en una ventana el código fuente.

Cada componente tiene una serie de eventos que se pueden asociar con el código, es decir, el programa podrá reaccionar ante cualquier evento ocurrido durante la ejecución (clic de ratón, tecla pulsada, mensajes recibidos, *timers*, etc.). El programador podrá definir de forma visual los eventos que contienen código en cada caso, lo que se traduce en la creación de una serie de métodos en el código fuente. En este instante se puede editar y completar su código.

Estas facilidades se agrupan en los diferentes componentes que se nos presentan:

1) *Componentes estándar de Windows*: Controles básicos, barras de herramientas, menús, botones animados, etc.
2) *Componentes de multimedia, bases de datos y Web.*
3) *Diálogos* de uso común.
4) *Form wizards*. Especialmente diseñados para automatizar los formularios de acceso a las bases de datos.
5) *Application wizards*. Nos permiten construir una sencilla aplicación en unos pocos minutos.
6) *Ayuda*. Proporcionan las herramientas necesarias para integrar la ayuda dentro de nuestra aplicación.

Tabla 3. Posibles Instalaciones

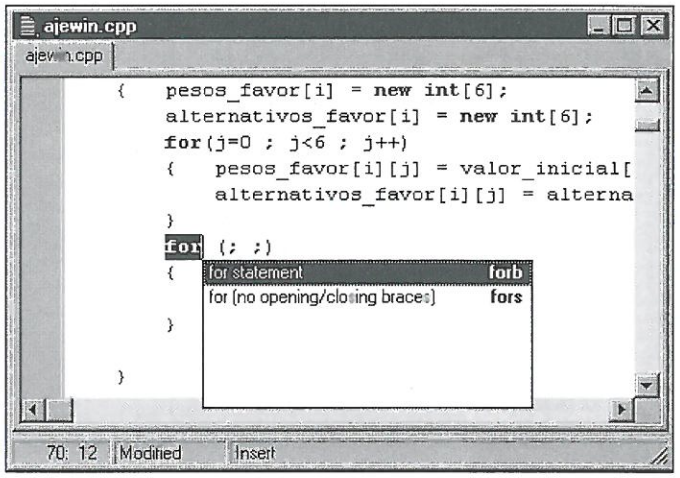
Opción	Espacio	Componentes
Típica	195 MB	No contiene ni OWL, ni MFC, ni MsHelp.
Mínima	128 MB	Sólo instala el motor de base de datos y el compilador C++ Builder con su entorno de desarrollo
Personalizada	--	--
Completa	270 MB	Todo

Mediante un asistente convertiremos un formulario en un componente ActiveX

- **Reutilización del Código.** A nivel de:
  - *Aplicación*. Una aplicación puede guardarse en cualquier estado y utilizarse como punto de arranque para el desarrollo de otra nueva.
  - *Cuadros de Diálogo* (formularios). Cualquier diálogo se puede guardar como componente para su posterior uso en otra aplicación. Esto nos va a permitir que sea muy sencillo para una organización crear y reutilizar formularios de entradas y análisis de datos. En este punto conviene mencionar el asistente que permite convertir un formulario en un componente ActiveX, que podrá utilizarse en aplicaciones de terceros, como puede ser Microsoft Office.
  - *Componentes*. Cualquier clase generada a partir de un componente perteneciente a la VCL puede convertirse en componente y añadirse a la paleta. Esto nos permitirá ajustarlos a nuestras necesidades reales.

También podremos convertirlos en controles ActiveX.

- *Código*. Proporciona un gran conjunto de librerías con funciones C y C++. Además, permite usar y generar plantillas de código para estructuras o sentencias, tales como **for**, **while**, etc., asegurándonos que la sintaxis será correcta. Esto se recupera mediante **Control + j**. Es posible estudiar las que contiene por defecto seleccionando **Opciones -> Code Insight -> Code Templates**.



Libre de Errores Sintácticos.

Todos los componentes existentes se podrán mantener de forma coherente gracias el *Object Repository*, que actúa como una especie de almacén controlado. Dentro del mismo se agrupan un conjunto de componentes, *wizards* (asistentes), aplicaciones y formularios, a partir de los cuales se puede empezar a desarrollar una aplicación. Va a permitir reducir mucho el tiempo de desarrollo, ya que tenemos pequeñas aplicaciones creadas que se pueden tomar como punto de partida.

COMPATIBILIDAD CON HERRAMIENTAS, CÓDIGO Y PLATAFORMAS EXISTENTES

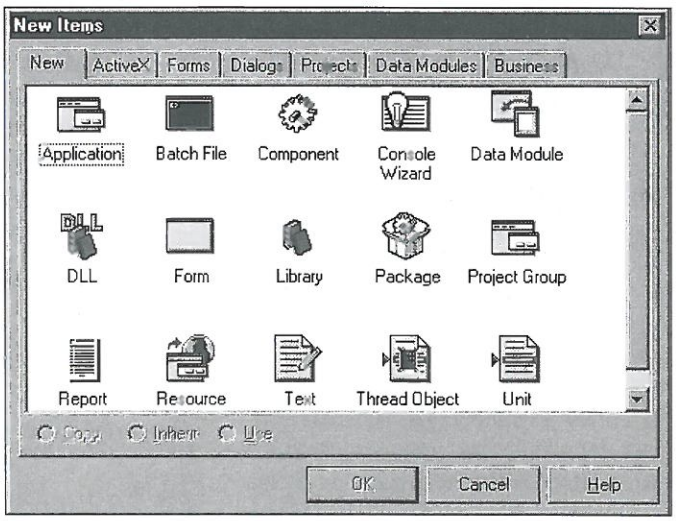
En la actualidad hay mucho código generado en C++ como para tener que volver a reescribirlo si queremos actualizar nuestro compilador. Así, C++ Builder 3 proporciona una compatibilidad total a diferentes niveles:

- *Sistema Operativo*. Genera código para Windows 95 y Windows NT. Además, es capaz de desarrollar *drivers* para estos sistemas. Por si fuera poco, Borland ha lanzado una versión especial de C++ Builder 3 que permitirá conseguir conectividad con las aplicaciones de AS/400.

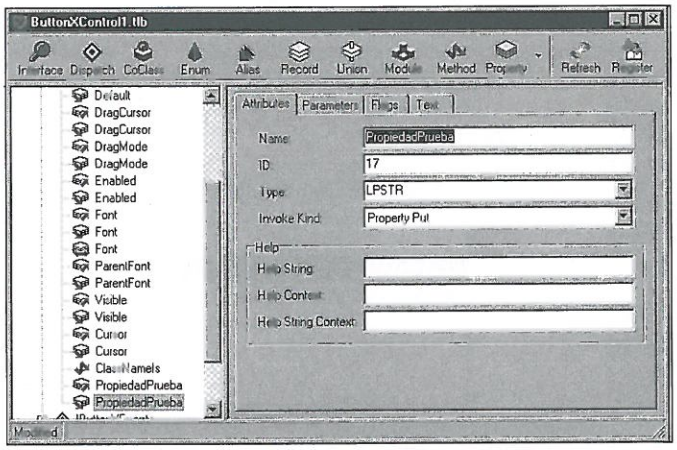
Puede trabajar con diferentes formatos de bases de datos y de servidores

- *Lenguaje*. Soporta antiguas versiones de C y C++, lo que facilita el uso de viejas librerías. También soporta las implementaciones de *OWL* y *MFC*. Además podremos seguir empleando nuestro código de Borland C++, en concreto, es capaz de convertir un proyecto de la versión 5.02 a partir de su *MAK*. Los ficheros de recursos, *RC*, también los convierte y genera los formularios asociados. Se hace mediante la opción Import Script RC. Esto último lo hemos probado y lo lleva a cabo bastante bien, lo único que era necesario tener el antiguo compilador, para usar los ficheros de cabecera.

- *Aplicaciones*. Por medio de DLLs, controles ActiveX y facilidades avanzadas de comunicación, C++ Builder 3 puede compartir código con otras aplicaciones como Delphi, Visual Basic, etc. Con respecto a la versión anterior de C++ Builder, hay que hacer unos pequeños cambios en la definición de los



Object Repository.



componentes, si queremos seguir usándolos en C++ Builder 3. Tendremos que añadir el identificador PACKAGE a todas las definiciones de las clases de nuestros objetos, así como a la función de Registro, y la directiva **#pragma package(smart_init)**.

- *Bases de Datos*. Puede trabajar con diferentes formatos y servidores. En el mismo entorno se hace el diseño de la aplicación, código C++, y el manejo de los datos, por medio de SQL. Las bases de datos que soporta son: ORACLE, Sybase, MsSQL Server, Informix, DB2, Interbase, dBase, Paradox, Access, FoxPro, ODBC y ASCII.

CARACTERÍSTICAS A DESTACAR

Innovaciones Tecnológicas.

- Creación de ActiveX a partir de los componentes y así poder integrarlos en cualquier aplicación. Para este fin tiene una utilidad que nos va a permitir mantener los ficheros *TLB*, definiendo las propiedades y los métodos asociados de una forma visual. Esto se refleja inmediatamente en el código fuente. Esta herramienta también registrará el nuevo control ActiveX creado. Ver figura 4.

- Controles orientados a la toma de decisiones. Manejar datos y presentarlos en forma de informe o gráficos, permitiendo tratarlos para estudiar su evolución.

- Reducción de los errores durante la fase de desarrollo, gracias a un generador de código que muestra la sintaxis de algunas de las sentencias que escribimos; y a la creación visual de componentes a partir de los ya existentes.

- Reducción de los ejecutables al separarlos en paquetes que se unirán en el momento de la ejecución. La forma en la que se van a partir lo va a indicar el programador durante la compilación. Los paquetes están orientados a la reducción del tráfico de red, el tiempo de compilación y de desarrollo, así como el facilitar el mantenimiento de las aplicaciones *n-tier* distribuidas.

Es posible realizar la reducción de los ejecutables al separarlos en paquetes

Depurador.

Se ha mejorado respecto a versiones anteriores. Este proporciona:

- *Visualización automática de las variables*, basta con colocar el cursor sobre la variable que se quiera inspeccionar, cuando estemos en modo de depuración, y aparecerá una *tool-tip* con el valor de esa variable.

- *Visor de módulos*: Muestra información detallada sobre los módulos utilizados por el proceso que estamos depurando.

- *Visor de eventos*, que muestra los mensajes de control, de puntos de ruptura o de ventanas.

- *Depurador integrado de DLLs*, que evita tener que utilizar programas adicionales para depurarlas.

Programación Integrada de Base de Datos.

C++ Builder 3 proporciona un desarrollo rápido de aplicaciones con bases de datos, gracias a:

- *Una arquitectura de bases de datos escalable y de alto rendimiento*. Puede usarse para cualquier tipo de aplicación, desde una simple base de datos, a una arquitectura cliente/servidor, o una aplicación distribuida a lo largo de una red local o extensa, incluyendo intranets e internets. La tecnología de bases de datos está estrechamente integrada en los componentes de C++ Builder 3. La arquitectura de bases de datos de C++ Builder 3 se basa en:

  1) Arquitectura completamente abierta y escalable.
  2) Soporte directo para la mayoría de las bases de datos.
  3) Soporte para el desarrollo de aplicaciones multi-tier cliente/servidor.

4) Soporte para la construcción y el desarrollo de bases de datos en la Web.
5) Acceso a bases de datos legales.

- *Componentes de bases de datos para crear aplicaciones visuales*. Estas componentes forman parte de la VCL, por lo que se van a poder tratar a un alto nivel, o modificar y personalizar, según las necesidades.

  Los componentes de las bases de datos se pueden agrupar en:

  1) Acceso a los datos.
  2) Visualización de los datos.
  3) Soporte para la toma de decisiones. Estos componentes visualizan los datos con un alto nivel de análisis de los mismos. Estos son:

     - *DecisionCube Crosstab*. Análisis complejo de los datos.
     - *TeeCharts*. Representa gráficamente grandes volúmenes de datos.
     - *Integrated QuickReports*. Informes personalizados de los datos tratados.

- *Potentes herramientas de gestión de bases de datos*, que nos van a hacer el desarrollo y la gestión de los datos mucho más sencillo.

  Entre las más destacadas mencionaremos:

  1) *SQL Builder*. Connstruye complejas sentencias SQL.
  2) *SQL y DBExplorer*. Permite inspeccionar, crear y modificar tablas, *queries* (consultas) , índices y alias sin programación.
  3) Database Dictionary. Asegura la coherencia de los datos., permitiendo fijar los valores máximos y mínimos, valores por defecto, etc. Todo ello sin necesidad de insertarlo en el código.

Gestión de Proyectos Complejos.

- *Gestión Avanzada de Proyectos*. Las aplicaciones C++, normalmente, están formadas por diferentes elementos: ficheros de cabecera, de código fuente, de recursos, imágenes, etc. Es más, la distribución de la aplicación final puede implicar ficheros *EXE*, *DLLs*, etc. Para facilitar esta tarea, C++ Builder 3 incluye un potente gestor de proyectos que representa gráficamente todas las dependencias. Figura 5. Así, mientras usamos el mismo proyecto, podemos desarrollar ficheros *EXE, LIB, RES* y *DLL*. Si se usan parámetros de compilación diferentes, se pueden obtener distintas variaciones de la misma aplicación. Así, de una única compilación, se puede generar un *EXE* en diferentes lenguajes.

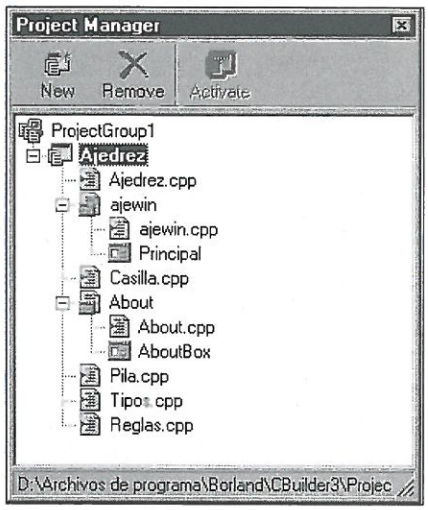
- *Gestión de Cambios*. Estos problemas se ponen de manifiesto cuando en el desarrollo están implicadas más de una persona. Afectan a dos áreas:

  1) *Control de Versiones*. Mantenimiento automático de ficheros de versiones anteriores.
  2) *Modificación de los fuentes*. Asegura que todos podrán estar viendo uno en particular, pero sólo uno podrá estar modificándolo a la vez.

Migración a la WEB

C++ Builder 3 presenta 25 componentes que proporcionan una fácil implementación de *sockets*, UDP, TCP, SMTP, POP 3, HTML, FTP, HTTP, y otros. El resultado es un entorno de desarrollo para la creación de aplicaciones orientadas a Internet de forma rápida y eficiente.

En el caso de tener un servidor Web, cuando generemos un control ActiveX se nos presentará la posibilidad de registrarlo en el mismo. De esta forma, las aplicaciones que lo utilicen lo descargarán de forma automática.



Visor de Proyectos.

CONCLUSIONES

Es importante resaltar que estamos ante una verdadera aplicación RAD (*Rapid Application Developement*) orientada a objetos. Con C++ Builder 3, el C++ no será sólo cosa de los programadores de sistemas, ya que ofrece una muy buena integración con diferentes sistemas de bases de datos, lo que la convierte en una herramienta muy a tener en cuenta. Por otro lado, el desarrollo de aplicaciones para Internet también a sufrido un fuerte impulso, ya que nos facilita algunas tediosas tareas. En otro orden de cosas, el entorno de desarrollo, lo han mejorado respecto a la versión anterior y dotado de las mejoras que ya traía la versión 3 de Delphi. Para finalizar, el proceso de compilación y generación de los ejecutables se ha visto acelerado de manera apreciable. Se trata en definitiva de una herramienta muy a tener en cuenta en nuestros futuros desarrollos, independientemente de la naturaleza de estos.

Para cualquier duda, crítica, sugerencia o demás se puede contactar con el autor por medio del e-mail: jaromero@openbank.es. Más información relacionada en www.borland.com.

LIBROS

Lotus Notes 4.x desarrollo de aplicaciones

Lotus Notes es el producto líder en el mercado mundial para el desarrollo e implantación de aplicaciones de mensajería y trabajo en grupo, permite al usuario incrementar su productividad y la cualidad en cuanto a toma decisiones.

Cualquiera que utilice Lotus Notes, ya sea programador o un simple usuario puede desarrollar potentes y precisas aplicaciones con este libro.

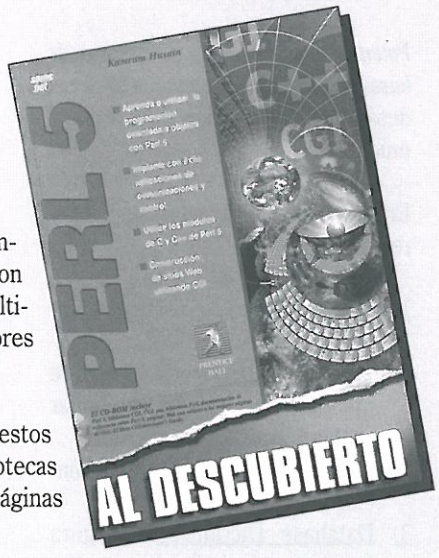
El objetivo fundamental del libro es que el lector comience a crear sus bases de datos Notes lo antes posible para lo cual se ha seguido una estructura de aprendizaje progresiva.

Se tratan a fondo la creación de formularios, vistas y agentes, la definición de campos y la especificación de fórmulas para la creación de potentes aplicaciones al servicio de Internet.

Es un libro muy didáctico, claro y conciso, lo que le hace ideal para el perfeccionamiento del profesional o para el aprendizaje del estudiante aunque la aplicación de que trata resulte de díficil comprensión.

Editorial: Mc Graw Hill — Autor: Javier Plaza
Nº páginas: 327 — Idioma: Español
Nivel: Todos los niveles — Precio: 3.700 Ptas.(I.V.A. inc.)

Perl 5 al descubierto



Perl es un lenguaje interpretado optimizado para recorrer archivos de texto arbitrario, extraer información de ellos e imprimir informes basados en esta información.

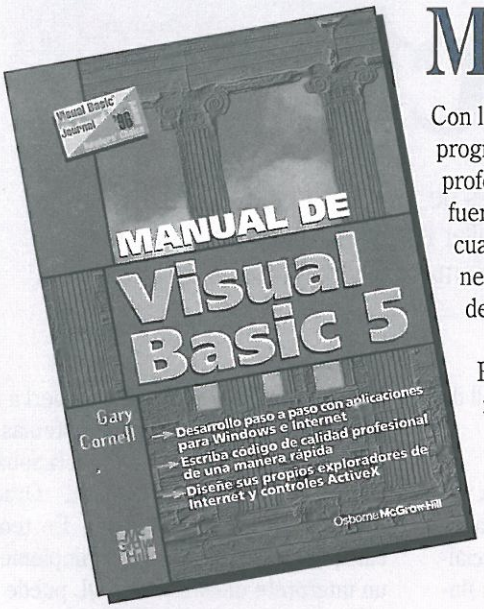
Con este libro el lector puede estudiar las alternativas Cliente/Servidor utilizando Perl, implementar funciones y estructuras, usar ventanas utilizando Perl/TK. Con este manual aprenderá a construir y explorar modelos VRML, desarrollar planos multidimensionales utilizando librerías GD, aprender a explotar valores escalares, operadores básicos y arrays asociativos.

La obra le proporciona una información exhaustiva sobre todos los temas expuestos anteriormente y además viene acompañado de un cd-rom que contiene Perl 5, bibliotecas documentación de referencia y numerosas páginas Web con enlaces a las mejores páginas de Perl.

Editorial: Prentice Hall — Autor: Kamram Husain
Nº páginas: 851 — Idioma: Español
Nivel: Experto — Precio: 7.500 Ptas.(I.V.A. inc.)

LIBROS

Manual de Visual Basic 5

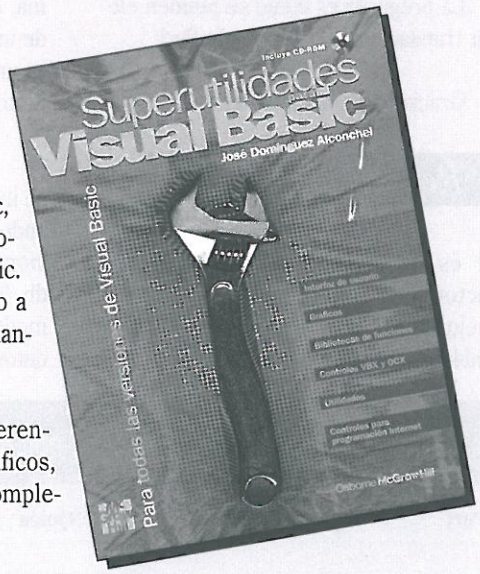

Visual Basic Journal
MANUAL DE
Visual Basic 5
Gary Cornell
Desarrollo paso a paso con aplicaciones para Windows e Internet
Escriba código de calidad profesional de una manera rápida
Diseñe sus propios exploradores de Internet y controles ActiveX
Osborne McGraw-Hill

Con la publicación de Visual Basic 5, Microsoft ha convertido lo que era el lenguaje de programación más popular en una auténtica herramienta de desarrollo de aplicaciones profesionales. Usado por millones de usuarios en todo el mundo, Visual Basic está fuertemente ligado a Windows e Internet, pudiendo compararse con Java y C en cuanto a funcionalidad. Para dominar este sofisticado entorno y competir con él, necesitará una guía y profesor experto como Gary Cornell, el autor de esta obra y de muchas otras sobre Visual Basic y otros lenguajes de programación.

Este Manual de Visual Basic 5 le llevará desde los fundamentos de programación más elementales a las técnicas actuales más avanzadas. A través de sus instrucciones modulares paso a paso avanzará rápidamente desde las bases de Basic a programar aplicaciones completas, e incluso a escribir componentes con los que ampliar programas realizados por otras personas.

Editorial: Mc Graw Hill
Nº páginas: 835
Nivel: Principiante y medio
Autor: Gary Cornell
Idioma: Español
Precio: 7.500 Ptas.(I.V.A. inc.)

Superutilidades para Visual Basic



Son muchos los libros que explican cómo aprender a programar con Visual Basic, a raíz de esto nace el libro que se comenta. Superutilidades para Visual Basic recoge técnicas, trucos y utilidades para programar eficientemente con Visual Basic. Encontrará multitud de ejemplos de programación, detallados y explicados paso a paso, y soluciones profesionales a los problemas más frecuentes que aparecen cuando se programa con Visual Basic.

Los temas sobre los que trata el libro son la programación en 32 bits, diferencias entre Windows 95, NT 3.51 Windows NT 4.0, interfaz del usuario, gráficos, multimedia, bibliotecas de funciones, controles VBX y OCX y utilidades complementarias para el buen desarrollo de la tarea del programador.

El libro va acompañado de un CD-ROM que incluye los ejemplos y código fuente del libro, código fuente de ejemplos adicionales, controles para programación en Internet, descompiladores, protectores de programas, tutorial de Visual Basic HTML (en inglés), demo de Visual Basic 4 y mucho más.

Este libro en definitiva es válido para todos los programadores de todas las versiones de Visual Basic.

Editorial: Mc Graw Hill
Nº páginas: 592
Nivel: Todos los niveles
Autor: José Domínguez Alconchel
Idioma: Español
Precio: 7.900 Ptas.(I.V.A. inc.)

Correo del lector

En esta sección, los lectores de SÓLO PROGRAMADORES tienen la oportunidad de hallar respuesta a los problemas que puedan tener en cualquier tema relacionado con la programación.

Pregunta

Ante todo disculpas si lo que voy a preguntar ya ha sido tratado en alguno de los números de SP, el caso es que me suena haberlo visto en algún lado pero no consigo encontrarlo, puede que fuera en alguna otra fuente.

La pregunta es cómo se pueden ejecutar transacciones SQL desde Perl.

Gracias por adelantado....

Respuesta

Perl es un lenguaje interpretado con características muy potentes para el manejo de cadenas, escalares y proceso de tablas desarrollado por Larry Wall creando un perfecto puente entre los shell de Unix y el puro y duro C.

Debido a que las operaciones en bases de datos relacionales son orientadas típicamente al texto, perl esta especialmente bien indicado para manejar los flujos de datos. Su código fuente que esta disponible de manera gratuita y que se puede ejecutar sobre cualquier plataforma, contiene un punto de acceso a partir de una función definida por el usuario que permite al desarrollador extender el conjunto de funciones básico del lenguaje.

Dentro del grupo de trabajo Dbperl, se intenta explotar esta característica creando un conjunto estandarizado de extensiones de funciones de perl (como db_fetch(), db_attach(), etc.) basadas en el modelo SQL para manipular una base de datos relacional, de manera que proporciona una interfaz portable para perl a una gran variedad de populares sistemas de manejo de bases de datos relacionales (RDMS) incluyendo Sybase, Oracle, Ingres, Informix, e Interbase. En teoría, cualquier base de datos que implemente un interprete dinámico de SQL puede ser utilizado por un *front-end* en perl.

La sede oficial en Internet para las extensiones Dbperl está en la dirección ftp://ftp.demon.co.uk/pub/perl/db. El archivo contiene copias de *ports* para Ingres, Oracle, Sybase, Informix, Unify, Postgres, and Interbase, asi como rdb and shql; es el hogar de la evolucionante especificación API de Dbperl.

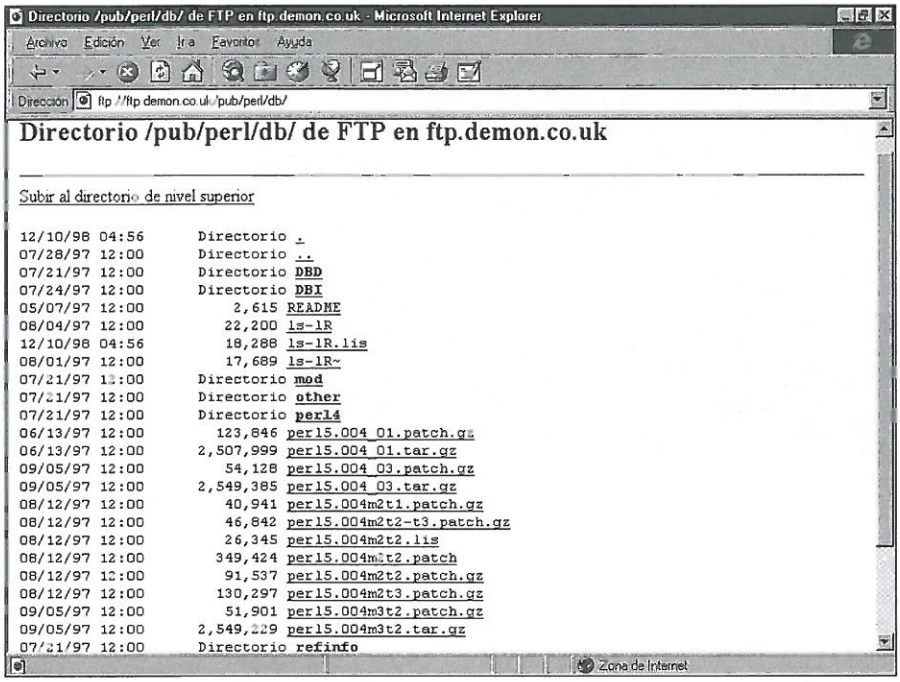
En el listado adjunto te mostramos una lista con los principales *ports* existentes, el DBMS al que hacen referencia y el autor de los mismos:

Lista con los principales ports

Port	DBMS	Quien
?Infoperl	Informix	Kurt Andersen (kurt@hpsdid.sdd.hp.com)
Ingperl	Ingres	Tim Bunce (timbo@ig.co.uk)
Interperl	Interbase	Buzz Moschetti (buzz@bear.com)
Isqlperl	Informix	William Hails, bill@tardis.co.uk
Oraperl	Oracle	Kevin Stock (kstock@Auspex.com)
Pgperl	Postgres	Igor Metz (metz@iam.unibe.ch)
*Sqlperl	Ingres	Ted Lemon (mellon@ncd.com)
Sybperl	Sybase	Michael Peppler (mpeppler@itf.ch)
Uniperl	Unify 5.0	Rick Wargo (rickers@coe.drexel.edu)

Pregunta

Hola!. Estoy diseñando un sitio Web y tengo una pregunta referente a los navegadores con soporte VRML. Se que puedo abrir una nueva ventana del navegador a través de un nodo ancla y mostrar un mundo VRML, pero, ¿es posible conectar de alguna manera los dos mundos VRML?. Esto me permitiría empezar una animación en el segundo navegador, si por ejemplo, seleccionara algún objeto del primero. Les agradecería me dijeran, en caso


Directorio /pub/perl/db/ de FTP en ftp.demon.co.uk - Microsoft Internet Explorer
Archivo Edición Ver Ir a Favoritos Ayuda
Dirección ftp://ftp.demon.co.uk/pub/perl/db/
Directorio /pub/perl/db/ de FTP en ftp.demon.co.uk
Subir al directorio de nivel superior
12/10/98 04:56 Directorio .
07/28/97 12:00 Directorio ..
07/21/97 12:00 Directorio DBD
07/24/97 12:00 Directorio DBI
05/07/97 12:00 2,615 README
08/04/97 12:00 22,200 ls-lR
12/10/98 04:56 18,288 ls-lR.lis
08/01/97 12:00 17,689 ls-lR~
07/21/97 12:00 Directorio mod
07/21/97 12:00 Directorio other
07/21/97 12:00 Directorio perl4
06/13/97 12:00 123,846 perl5.004_01.patch.gz
06/13/97 12:00 2,507,999 perl5.004_01.tar.gz
09/05/97 12:00 54,128 perl5.004_03.patch.gz
09/05/97 12:00 2,549,385 perl5.004_03.tar.gz
08/12/97 12:00 40,941 perl5.004m2t1.patch.gz
08/12/97 12:00 46,842 perl5.004m2t2-t3.patch.gz
08/12/97 12:00 26,345 perl5.004m2t2.lis
08/12/97 12:00 349,424 perl5.004m2t2.patch
08/12/97 12:00 91,537 perl5.004m2t2.patch.gz
08/12/97 12:00 130,297 perl5.004m2t3.patch.gz
09/05/97 12:00 51,901 perl5.004m3t2.patch.gz
09/05/97 12:00 2,549,229 perl5.004m3t2.tar.gz
07/21/97 12:00 Directorio refinfo
Zona de Internet

Esta es la sede oficial para las extensiones Dbperl.

afirmativo, cómo realizarlo, pues me interesa incluir esta característica en mi proyecto.

Respuesta

Sí, es posible, y la manera más sencilla de realizarlo es usando un poco de programación Java. Necesitas un plug-in de tipo VRML 2 con soporte para EAI (External Authoring Interface), el cual permitirá a los applets Java controlar la aplicación VRML, estableciendo un enlace entre las dos sesiones del navegador. Puedes utilizar para ello tanto la versión 1.0 como la 2.0 del programa Cosmo Player.

Pregunta

Buenas. Tengo una pregunta sobre Linux que me está trayendo de cabeza. Acabo de actualizar mi CD-ROM 6x a 24x y Linux ya no me lo reconoce bien. Ahora, cuando intento montar el CD, obtengo un mensaje que dice *"mount: /dev/cdrom is not a valid block device"*. El nuevo CD es una unidad IDE/ATAPI y estoy usando RedHat 5.0. He probado incluso a reinstalar todo, siendo muy extraño puesto que puedo arrancar en DOS e instalar la RedHat sin problemas directamente del CDROM. Pero después, cuando reinicializo y arranco Linux obtengo el error de nuevo. ¿Qué está mal?

Respuesta

Lo más seguro es que sea un problema de la localización de tu unidad lectora de CDs. Bajo Redhat 5.0, /dev/cdrom es un enlace a /dev/hdc. Esto significa que debería ser la unidad master de tu controladora secundaria. Intenta montar /dev/hdb en /dev/hdd y prueba si así funciona.

Pregunta

Hola. Estoy ejecutando Slackware Linux 2.0.30 en un 486 DX2 y tengo un pequeño problemilla con las comunicaciones a través de la utilidad getty. En el puerto com2/ttyS1 tengo un módem 28.800. Me gustaría hacer que Linux respondiera llamadas con un prompt. Desafortunadamente, no consigo que responda al teléfono, y cada 5 minutos más o menos, obtengo un mensaje tal que así

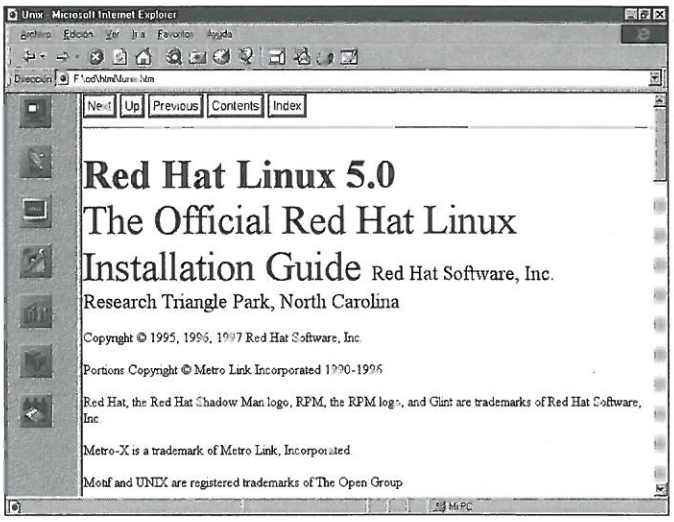
"d2 is respowning too fast and is disabled for 5 minutes". Tengo descomentada la línea d2 en el fichero /etc/inittab, la cual contiene lo siguiente:

d2:12345:respawn:/sbin/agetty –mt60 38400,12900,9600,2400,1200 ttyS1 t100

¿Que significa esto?.

Respuesta

Esto significa que tu módem tiene el echo activado y todo lo que le envías lo devuelve de nuevo al getty. Para solucionarlo,


Unix - Microsoft Internet Explorer
Archivo Edición Ver Ir a Favoritos Ayuda
Next Up Previous Contents Index
Red Hat Linux 5.0
The Official Red Hat Linux
Installation Guide Red Hat Software, Inc.
Research Triangle Park, North Carolina
Copyright © 1995, 1996, 1997 Red Hat Software, Inc.
Portions Copyright © Metro Link Incorporated 1990-1996
Red Hat, the Red Hat Shadow Man logo, RPM, the RPM logo, and Glint are trademarks of Red Hat Software, Inc.
Metro-X is a trademark of Metro Link, Incorporated.
Motif and UNIX are registered trademarks of The Open Group
Mi PC

Dentro del CD-ROM de Linux Red Hat 5.0 se puede encontrar ayuda on-line.

sólo tienes que desactivar el echo local de tu módem. En un USR bastaría con **"ATE0F1"**.

Pregunta

Hola amigos. Recurro a vosotros porque no consigo hacer arrancar Linux desde el disco duro (utilizo la distribución Slackware). En su momento instalé y configuré el programa Lilo que me comentó un amigo para dejar de depender del disquete de arranque, pero el ordenador se me queda colgado nada más encenderlo y ahora tengo que usar un disco incluso para acceder al Windows 95.

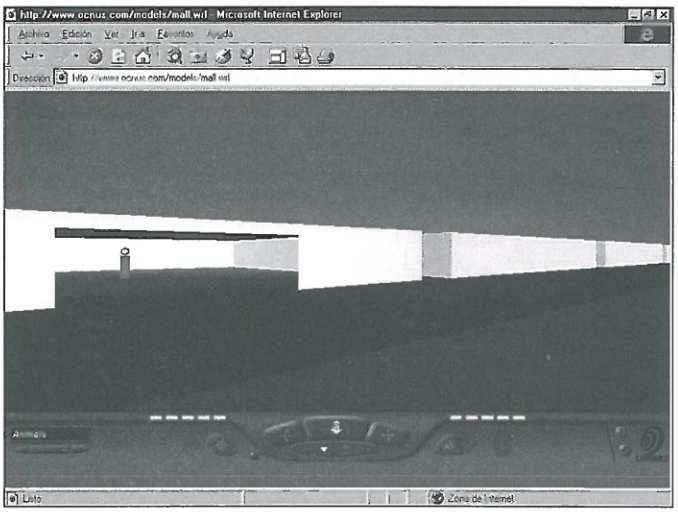
Nada más sucederme esto me asusté y como suelo tener un backup de toda la información crucial, lo que hice fue reformatear el disco duro para volver a instalar el Windows 95 que ya no me arrancaba. La cosa es que ni aún así se ha solucionado, puesto que me aparecen unos símbolos extraños en el momento de arrancar el ordenador.

En estos ultimos días no he necesitado mucho el ordenador y el amigo que me ayuda con él está de viaje en estos momentos. ¿qué tengo que hacer para eliminar todo el desaguisado que me ha hecho Linux?. Realmente, estoy un poco asustado.

Respuesta

Bien, ante todo calma. No hay problema que no tenga solución, y el tuyo es de lo más sencillo. Cierto es que todos tenemos un miedo interior cuando nos enfrentamos con cosas nuevas, pero errando es como más aprende uno, aunque lamentamos que estos días hayas tenido que prescindir de tu computadora para esperar nuestra solución.

El programa que instalaste, **Lilo**, no es más que un gestor de arranque de lo más versátil que posee Linux y que hace de él un sistema operativo que puede integrarse perfectamente con otros careciendo de cualquier incompatibilidad. Por lo que deduzco, cuando instalaste Linux, lo hiciste en una partición de un disco duro con más de 1024 cilindros.



Gracias a los plug-in de tipo VRML 2 con soporte para EAI es posible que los applets Java controlen la aplicación.

Es un pequeño fallo de Lilo, puesto que no reconoce más allá del cilindro 1024, y no puede encontrar los ficheros de arranque de Linux que necesita (**vmlinuz** y **boot.b**). Seguramente lo que te aparecería al arrancar entonces sería una *"L"* y se te quedaba colgado, en lugar del acostrumbrado mensaje *"Linux Boot:"*.

Para solucionar esto, y ya que has reformateado las particiones y te has cepillado todo, te aconsejamos que reparticiones de nuevo y localices la partición que quieras usar para Linux dentro del primer Gyga del disco duro en cuestión.

Para eliminar todas las trazas que pudieran dejar los anteriores sistemas operativos y dejar el ordenador limpio para formatear e instalar un nuevo S.O., usa un disco de arranque de MS-DOS 5 o superior o el propio disco de arranque de Windows 95 y ejecuta la orden: **"fdisk /MBR"**, esto hará que el programa fdisk borre completamente el *Master Boot Record* de tu disco duro.

Ahora ya podrás proceder con la instalación normal del Windows 95 y luego puedes hacer lo propio con Linux, te aseguramos que ya no tendrás ningún problema parecido.

Si nos hubieras escrito antes de borrar los discos, te habríamos dado como solución para arrancar Linux, que copiases ambos ficheros mencionados anteriormente (boot.b y vmlinuz) a cualquier partición DOS que tuvieras dentro del rango 0-1024 cilindros del disco duro y modificases un poco el fichero de configuración de Lilo **lilo.conf** añadiendo dos líneas:

```
install = /****/boot.b
map = /****/map
```

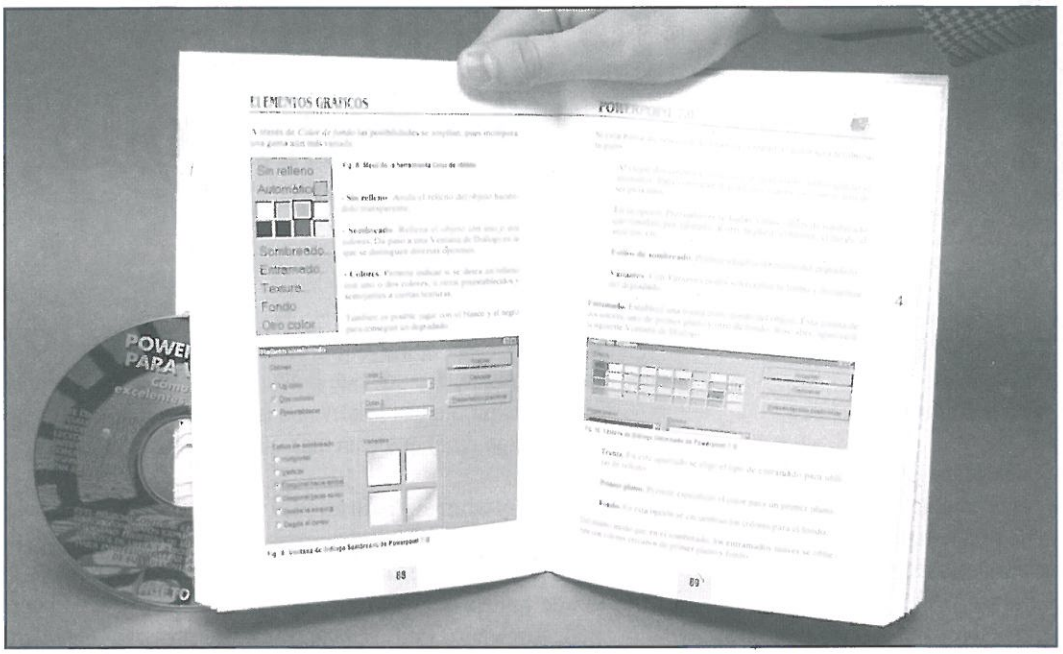
y modificando la localización de la imagen, donde los asteriscos corresponden al nombre que le has dado a la partición DOS al montarla en Linux. (por supuesto, es necesario tener acceso desde Linux a la partición DOS):

```
image = /***/vmlinuz
```

Hay una tercera opción para arrancar Linux directamente sin disco, y es hacerlo a través de una utilidad llamada **Loadlin**. Al igual que antes, copiamos la imagen del kernel (**vmlinuz**) a nuestra partición DOS y luego ejecutamos esta utilidad desde el propio MS-DOS, o lo configuramos como una opción del menú de arranque de MS-DOS o Windows 95, añadiendo las correspondientes líneas al fichero autoexec.bat.

La mayoría de anuncios editoriales te muestran los libros cerrados

NOSOTROS HEMOS PENSADO QUE PREFIERES VERLOS ABIERTOS



Libro + CD-ROM
por sólo
1.995
ptas.
IVA incluido

- ✓Explicaciones paso a paso
- ✓Ejemplos prácticos a todo color
- ✓Imágenes de cada una de las pantallas que encontrarás en tu ordenador

Biblioteca de Informática AVANZADA

Títulos que componen la colección "Biblioteca Informática Avanzada":

- EXPLORANDO INTERNET
- 3 D STUDIO V.4
- AUTOCAD 13
- MÚSICA DIGITAL POR ORDENADOR
- REALIDAD VIRTUAL
- LENGUAJES HTML, JAVA Y CGI
- WORD PARA WINDOWS 95
- COREL 6.0
- VISUAL BASIC 4.0
- POWERPOINT 7.0 PARA WINDOWS 95
- EXCEL 7.0 PARA WINDOWS 95
- JAVA. EL LENGUAJE DE INTERNET
- MICROSOFT WORD 97
- MICROSOFT ACCESS 97
- MICROSOFT EXCEL 97

Distribuidores autorizados en librerías

COMUNIDAD DE MADRID / CASTILLA LA MANCHA		**CANARIAS**	
DISTRIFORMA S.A.	Tfno. 91 - 501 4749	GARCIA PRIETO LIBROS S.L.	Tfno. 922-820026
CATALUÑA		**BALEARES**	
MIDAC LLIBRES S.L.	Tf. 93-421 18 95	PONENT LLIBRES S.L.	Tfno. 971-430339
ANDALUCIA OCCIDENTAL/ EXTREMADURA		**VALENCIA - CASTELLÓN**	
DISTRIBUCIÓN DE EDICIONES RDGUEZ. SANTOS S.L.	Tfno. 95 - 418 04 75	ADONAY S.L.	Tfno. 96-3793151
ANDALUCIA ORIENTAL		**ALICANTE - MURCIA - ALBACETE**	
DISTRIBUCIONES DEL MEDIODIA S.A. (ZÓCALO)	Tfno. 958-550278	LA TIERRA LIBROS S.L.	Tfno. 96-5110192
CASTILLA - LEÓN		**PAÍS VASCO - NAVARRA**	
ARCADIA S.L.	Tfno. 983-395049	YOAR S.L.	Tfno. 948-302239
GALICIA		**ARAGÓN - LA RIOJA**	
LUIS REY ABELLA (DISGALIBRO)	Tfno. 981-795754	ICARO DISTRIBUIDORA S.L.	Tfno. 976-126333
ASTURIAS - CANTABRIA			
DISTRIBUCIONES CIMADEVILLA S.A.	Tfno. 98-5167930		

ABECE c/ Aragoneses, 7. 28.108 Alcobendas (MADRID)
Tel.: (91) 661 42 11* Fax: (91) 661 43 86

CONTENIDO CD-ROM

Las novedades aparecidas en el mercado del software de desarrollo

PROGRAMACIÓN

- **ACUCOBOL-GT 3.2** permite a los programadores:

  Implementar y distribuir aplicaciones COBOL con un agradable interfaz de usuario que incluye: ventanas flotantes, controles gráficos, campos de texto, marcos, botones circulares, botones pulsables y etiquetas, barras de menú, bitmaps y barras de herramientas.
  Portado de aplicaciones sobre más de 600 plataformas usando un único conjunto de instrucciones.
  Actualizar programas que cumplan con el estándar previo ANSI-85 para programas en COBOL.

  ACUCOBOL-GT es un compilador de una sola pasada. Una compilación con éxito, produce un ejecutable que es posible arrancar sobre cualquiera de las plataformas antes mencionadas. El programa no necesita ser linkado para poder ser ejecutado.

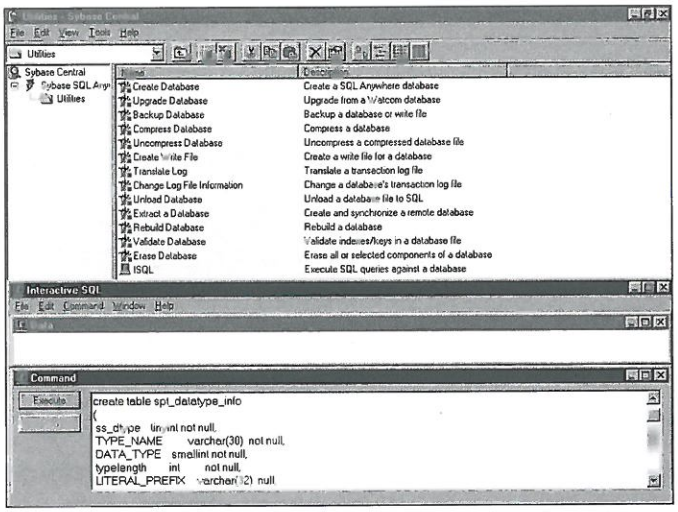
- Sybase (Powersoft) lanza la última versión de su herramienta profesional para la creación de aplicaciones de negocios que incorpora una fábrica de componentes (Component Factory) para la distribución de objetos **PowerBuilder** como si fuesen componentes estándar de la industria del software.

  PowerBuilder también soporta pequeños clientes Web, disponibilidad en plataformas Unix y mejoras para el desarrollador que necesite crear aplicaciones basadas en componentes distribuidos. Estos podrán ser fácilmente implementados en toda una gama de arquitecturas. PowerBuilder v6.0 expande las capacidades de generación del HTML dinámico para clientes Web muy pequeños y conectividad de alto rendimiento para clientes Java. Además presenta un entorno seguro de mantenimiento mínimo para la ejecución de componentes cliente PowerBuilder que se descargan del servidor. Estas características dan a los desarrolladores la flexibilidad de crear aplicaciones Web utilizando las arquitecturas existentes, dejando que PowerBuilder se ocupe de solucionar los problemas relativos a las transacciones de negocios, campo en el que está más que probada su eficacia.

- **SQL Anywhere Professional** es un DBMS "Database Management System" que extiende la intranet corporativa a cualquier lugar (anywhere) donde trabajemos, o donde quiera que ocurran las transacciones comerciales. SQL Anywhere Professional introduce mejoras tecnológicas para posibilitar a las organizaciones desarrollar contenido Web dinámico. SQL Remote suministra capacidades optimizadas de replicación para usuarios esporádicos. Esto es muy importante en aplicaciones donde un gran número de usuarios remotos se conectan periódicamente al servidor para realizar sus transacciones y sincronizar sus bases de datos locales con las de éste. NetImpact Dynamo permite a los desarrolladores el uso de plantillas o páginas HTML con órdenes y scripts SQL en su interior. Estas plantillas se crean mediante potentes Wizards, que incluso un novato en el uso del lenguaje de consulta a bases de datos puede seguir. Además incluye sedes Web de ejemplo, una galería de estas plantillas y un lenguaje integrado de script totalmente compatible con JavaScript para ayudar en la creación de aplicaciones Web completas.

- **IFilter** es un interfaz OLE que permite a los desarrolladores escribir filtros de contenido personalizado para sus propios formatos de datos. Microsoft Index Server usa estos filtros de contenido para extraer texto de formatos de datos privados. De hecho, Index Server proporciona filtros para documentos con formato de texto, HTML, Microsoft Word, Microsoft Excel y Microsoft PowerPoint. Con el interfaz IFilter, los programadores pueden crear sus propios filtros de contenido para los formatos de datos de sus aplicaciones.

- **TAPI 2.1** está ahora disponible para desarrolladores. Si está desarrollando una aplicación que hace uso de los servicios



Las aplicaciones de Sybase son las estrellas del CD-ROM de este mes.

de telefonía de Microsoft Windows 95, Microsoft Windows NT 4.0 Workstation o Server, este Software Development Kit le será de gran utilidad.

REDES

- **Apache** ha sido el servidor más popular en Internet desde abril de 1996. Según recientes estudios, más del 50% de las sedes Web en Internet usan Apache y sus derivados. El Apache Proyect es el fruto de un esfuerzo para desarrollar y mantener un servidor HTTP con fuentes públicas para varios entornos modernos y sistemas operativos, orientados a servir como Unix y Windows NT. El verdadero éxito de este proyecto es proporcionar un seguro, ampliable y eficiente servidor que nos facilite servicios HTTP en sintonía con los actuales estándares en hipertexto. Desde que la Red de Area Global (WWW) afecta tanto a una plataforma que sirve documentos estáticos, como a otra que ejecuta aplicaciones dinámicas en diferentes plataformas de forma distribuida, los servidores HTTP (HyperText Transfer Protocol) juegan un papel cada vez más relevante. Este servidor te permite crear y desarrollar ese tipo de aplicaciones.

Proporciona un marco de trabajo que acompaña con una arquitectura modular distribuida y una API pública que permite al usuario crear programas portables en diferentes paradigmas de programación.

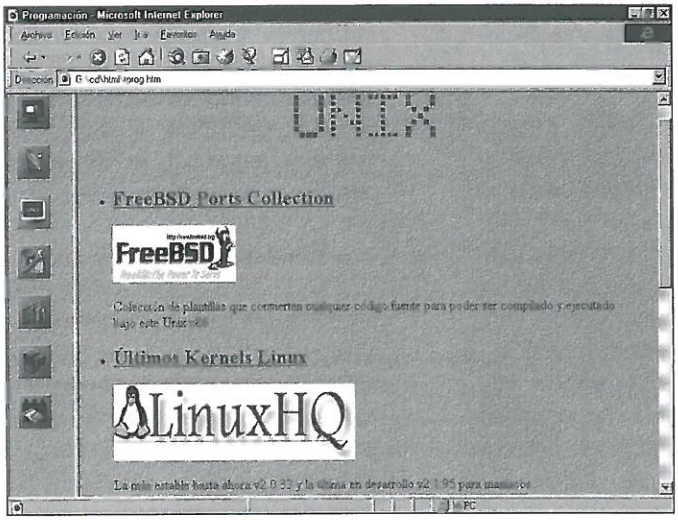
El avance que están experimentando los negocios basados en la Web, hacen imprescindible para el desarrollador el conocimiento de mucho más que una aplicación de diseño cliente/servidor y su técnica. Los desarrolladores necesitan conocer miriadas de tipos de datos, componentes, protocolos, metodologías push, actualizaciones dinámicas y aplicaciones que incorporen la tecnología de, podríamos decir, la semana pasada. Ahora, la compañía que trajo al mundo PowerBuilder, entorno específico para un desarrollo cliente/servidor altamente productivo, define una nueva categoría en la producción de aplicaciones Web con PowerSite, un conjunto de herramientas para la construcción, manejo y desarrollo de aplicaciones basadas en la Web para la empresa.

- **Active Server Pages** es una característica de los siguientes servidores de Web:
  - Microsoft Internet Information Server 3.0 para Windows NT 4.0
  - Microsoft Peer Web Services 3.0 para Windows NT Workstation 4.0

También puede instalar y usar Active Server Pages con Microsoft Personal Web Server version 1.0a para Windows 95, de manera que podrá crear aplicaciones ASP para depositar en Internet Information Server.

UNIX

- La colección de **Ports FreeBSD**, ofrece un camino para usuarios y administradores a fín de que puedan instalar aplicaciones en este popular sistema Unix para x86 de creciente éxito. Cada port, contiene los parches necesarios para convertir el código fuente de la aplicación original para



A través del interfaz HTML es posible acceder a las colecciones de elementos UNIX.

que se compile y ejecute perfectamente en FreeBSD. Instalar una aplicación es tan fácil como sacarla de Internet (o de este CDROM), descomprimirlo, y teclear make en el directorio del port. El fichero Makefile ajusta automáticamente el código fuente de la aplicación, ya sea desde un disco local, un CDROM o vía ftp desde Internet, aplica el parche y lo compila. Si todo ha ido bien, simplemente teclearemos make install para instalar la aplicación y registrarla en el sistema.

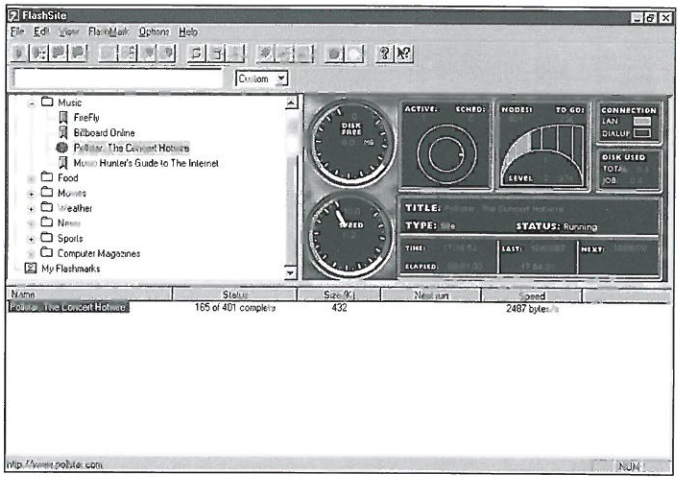
- **XEmacs** es un potente y extensible editor con total soporte gráfico, inicialmente basado en la antigua versión GNU Emacs 19 proveniente de la Free Software Foundation. XEmacs surge de la colaboración de Lucid Inc. con Sun Microsystems Inc. y la Universidad de Illinois.

- **Ygl** emula las rutinas de Silicon Graphics GL bajo X11. Compila perfectamente en AIX 3.2, HP-UX 7.0/8.0/9.0, Linux con XFree, SunOS, ConvexOS y otros, pero necesita un compilador ANSI-C (con gcc de GNU es suficiente).

VARIOS

En este apartado hacemos especial mención a las herramientas de navegación en diferido (off-line) incluyendo tres de las más conocidas:

- **FlashSite v1.01** permite al usuario moverse de forma transparente entre los datos remotos y locales al tiempo que se navega por la red. Posee una barra flotante que permite acceder a la mayoría de las funciones al mismo tiempo que usamos un navegador convencional. Permite además mapear una sede Web sin que haya necesidad de descargarla entera en nuestro ordenador. Podemos definir los límites de la descarga tanto por la profundidad de los enlaces como por la cantidad de disco ocupado. Ofrece además la posibilidad de programar las descargas, así como la generación de informes y las posibles actualizaciones necesarias cuando una página cambie.



FlashSite es uno de los programas más potentes para realizar navegación off-line. Permite guardar sites Web completos en el disco duro local.

- **Web Buddy v1.1** instala una barra flotante que se lanza automáticamente al lanzar Netscape o IE. Cuando deseemos guardar una página en disco, haremos click en uno de los iconos de esta barra y automáticamente serán almacanados los gráficos, sonidos, multimedia y demás objetos de la página. De esta manera también podemos bajarnos una sede Web entera o programar a Web Buddy para que lo haga en los intervalos adecuados. Después de volcar una sede, podemos utilizar el panel de control de Web Buddy para navegar por él, lanzándolo en cualquier momento con nuestro navegador. Podemos además convertir las páginas a un formato de procesador de textos y programar las descargas desde el panel de control.

- Con **WebWhacker v2.5**, otra herramienta de navegación en diferido, además de las características esenciales que ofrecen las otras dos, permite ordenar nuestros criterios de almacenamiento de la información de acuerdo a los temas que nosotros elijamos, guardando nuestros downloads en diferentes apartados, para su posterior visualización. Además, permite realizar búsquedas booleanas de toda la base de datos de una forma bastante potente, haciendo que la localización de la información sea muy eficiente.

  El único problema que presenta esta trial version es el número de niveles que permite visitar al navegar, ya que está limitado a dos.

También hemos incluido lo último de Microsoft para aumentar el redimiento de Windows:

- El equipo de desarrollo de Microsoft ha puesto juntos un conjunto de utilidades de productividad denominadas **PowerToys y KernelToys**. Si usted es un usuario avanzado de Windows, puede instalarlas y sacar partido de ellas. Se ha puesto gran cuidado en asegurar que todas las herramientas de los paquetes PowerToys y KernelToys funcionen como es debido. Pero tenga en cuenta que estas utilidades no forman parte de Windows y no están soportadas por Microsoft. Por esta razón, el soporte técnico de Microsoft no responderá preguntas acerca de las mismas.

- **More Properties**, es una pequeña aplicación que le permitirá configurar características de Windows 95 y NT que no están accesibles desde el entorno original de ambos sistemas.

Red Hat Linux Eurielec 98 en castellano

CONTENIDO CD-ROM ADICIONAL

MANUAL DE INSTALACIÓN E INICIACIÓN

Este mini-manual, pretende ser una ayuda para poder instalar Red Hat en tu ordenador. Está pensado para gente que NUNCA ha instalado Linux. Si ya lo has instalado alguna vez, seguro que no tendrás ningún problema.

FUNDAMENTOS BÁSICOS

Cómo organizar el disco duro

El primer requisito para poder instalar Linux es tener bastante espacio libre en el disco duro. Este sistema operativo necesita dos particiones especiales para él sólo (aunque se puede hacer de otra manera, ésta es la más conveniente).

Si tienes una partición libre, o puedes eliminar una que ya tengas, no hay ningún problema, puedes saltar hasta el apartado 3, siempre y cuando la partición tenga el tamaño suficiente.

Si este no es tu caso, tendrás que reducir el espacio de tu partición de Windows 95 o MS-DOS, para conseguir liberar espacio para nuevas particiones.

Cómo reducir el tamaño de una partición

El primer paso para reducir el tamaño de una partición, consiste en desfragmentarla usando el programa **defrag**, que viene con Windows 95 y MS-DOS. El siguiente paso consistirá en crear un disco especial denominado *FIPS*, con el que se arrancará el equipo y se ajustará el espacio de las particiones existentes. Para crear el disco *FIPS* debes seguir los pasos siguientes:

a) Introduce un disco formateado en la unidad A:
b) Desde una ventana de MS-DOS escribe format a: /s
c) Copia los siguientes archivos del directorio **dosutils** del CD-ROM:

FIPS.EXE
RESTORRB.EXE
ERRORS.TXT

A continuación, reinicia el ordenador con el disco FIPS en la unidad A: (Asegúrate de que tienes activado el arranque desde la disquetera) y después ejecuta el programa **fips.** Este programa permite reducir el tamaño de una partición sin necesidad de formatearla, previa copia de seguridad de la información del disco. La posibilidad de reducirla dependerá del espacio libre disponible en la misma, y de los sectores "inamovibles".

El programa **fips** hará algunas comprobaciones en tu sistema, y te preguntará si deseas hacer una copia de seguridad del sector de arranque del disco. En este caso se debe responder de forma afirmativa. Más adelante te preguntará si tienes un disco de arranque en la unidad A:, siendo la respuesta igual que en el caso anterior. (El programa guarda una copia de seguridad, que se podrá restaurar mediante el programa **restorrb.exe**).

A continuación mostrará las particiones existentes para que elijamos la que vamos a "recortar" con las flechas arriba y abajo. Para saber el espacio necesario para la instalación y ejecución correcta de Linux consulta la siguiente sección. Por último, confirmaremos la selección (con **c** primero y con **e** después).

El espacio necesario para Linux

Linux necesita una partición de *swap* (intercambio) de unos 32MB (más o menos el doble de la RAM instalada) y una partición para instalar Linux en sí. El tamaño necesario para ésta dependerá de la cantidad de aplicaciones que deseemos instalar junto con el sistema:

a) Completo: 500Mb
b) Para programar y entorno gráfico:..... 350Mb
c) Básico, el entorno gráfico y poco más:.. 200Mb

EJECUTAR EL PROGRAMA DE INSTALACIÓN DE LINUX RED HAT

El programa de instalación se puede ejecutar directamente desde el CD-ROM, pero a veces se necesita crear los disquetes de arranque. Pueden darse varios casos:

a) **Si ya tienes Linux**
Crea los discos con:

```
dd if=/mnt/cdrom/images/boot.img
of=/dev/fd0
dd if=/mnt/cdrom/images/supp.img
of=/dev/fd0
```

b) **Si tienes MS-DOS**
Ejecuta el programa **autoboot.bat** del directorio **dosutils** (el archivo **Linux.bat** no funciona)

c) **Si tienes Windows 95 y soporte para CD-ROM en modo MS-DOS**
Si éste es tu caso (lo sabrás, si al reiniciar en modo ms-dos, el ordenador aún puede leer el cd-rom), reinicia en modo MS-DOS el sistema, y ejecuta el programa **autoboot.bat** del directorio **dosutils.**

d) **Si tienes Windows 95 sin soporte para CD-ROM en modo MS-DOS**
Introduce un disco vacío en la unidad A: , y desde una ventana de MS-DOS, ejecuta el programa **rawrite.exe** que viene en el directorio **dosutils** del CD-ROM, y cuando te pregunte por las imágenes, elije el fichero **boot.img** del directorio **images** (p. ej. **d:\images\boot.img**). A continuación reinicia el ordenador con el disco recién creado en la unidad A:

A partir de aquí, el programa de instalación te irá guiando paso a paso hasta la conclusión del proceso. Un momento decisivo de la instalación es el particionado de discos, que se llevará a cabo con el programa **disk druid** de la instalación.

Para Linux necesitarás crear dos particiones, una Linux nativa, y otra de *swap*. Si ya tenías una partición que deseas borrar, con el programa **disk druid**, elíjela, y bórrala. A continuación crea una partición de swap de unos 32 Mb. Utiliza el botón añadir y la selección en Linux swap para realizar esta tarea. Luego se debe crear la partición de Linux nativa, de nuevo con el botón añadir, y la selección en Linux nativa. El destino de montaje para esta partición será /

El resto de las fases de la instalación no presenta ningún problema. Sólo un detalle, cuando estés en la configuración de las X-Windows, no elijas la opción **autodetectar** los modos de vídeo, porque si lo haces no quedarán bien los caracteres típicos españoles como á é í ó ú, y eso es muy desagradable, ¿verdad?

Cuando tengas que instalar el boot loader (**LILO**), podrás elegir varios destinos para instalarlo, te recomendamos que elijas el MBR (*Master Boot Record*). De etsa forma LILO te permitirá elegir arrancar tu ordenador con Linux o con otro sistema operativo con sólo poner una palabra.

Cuando arranques por primera vez Linux, pon como *login* la palabra **root**, introduce el *password* y a continuación crea un usuario, con el comando:

```
adduser usuario
```

y luego ponle un *pasword*, diferente al del *root*, con el comando:

```
passwd usuario
```

A partir de ahora recomendamos que utilices Linux con ese usuario que acabas de crear, y sólo entres como root cuando quieras cambiar algo de la configuración del sistema, o instalar un determinado paquete.

PRIMEROS PASOS CON LINUX

Aquí detallamos algunas de los puntos que se deben seguir después de haber instalado *Red Hat*. Si tienes alguna duda consulta el apartado TRUCOS, que explica cómo realizar algunas de las tareas básicas.

Cómo instalar los upgrades

Las actualizaciones de Red Hat hasta la fecha se encuentran en el directorio **upgrades**. Recomendamos su instalación por motivos obvios (rpm). Existen nuevas actualizaciones disponibles en la web de redhat http://www.redhat.com/errata/errata.html y también en la web de EURIELEC RedHat en http://www.etsit.upm.es/~eurielec/redhat/.

Instalar los scripts de Red Hat Eurielec 98:

Estos se encuentran en el directorio **/scripts/**

¿Qué es EURIELEC?

Eurielec es una asociación de la ETSI de Telecomunicación de Madrid creada en 1970. Sus miembros son principalmente estudiantes de la escuela, con inquietudes en el campo de la informática y la electrónica.

Todo los años se organizan cursillos e intercambios con otras universidades europeas.

Varios miembros son aficionados a la programación y administración de sistemas (Linux y otros Unix's). De estos miembros nació la idea de crear una distribución de Linux en castellano pensada sobre todo para los usuarios noveles, que no habían tenido contacto con Linux. Por eso se pensó utilizar como base Red Hat, la que mejor combina la sencillez de instalación con la potencia.

a) Acceso a redes: Sirve para realizar la conexión a redes como en el caso de Internet. Está formado por varios *scripts* con el fin de facilitar la configuración.
b) Se instala desde el CD-ROM en el directorio **scripts/internet/**

Tras la instalación se ejecuta automáticamente el *script* de configuración del acceso a redes.

El primero consiste en crear los ficheros necesarios seleccionando la opción a. Luego hay que añadir un perfil de módem (se pueden añadir varios también) (opción b). Una vez creado al menos un perfil de módem se pueden configurar las conexiones. Puedes realizar varias y gestionarlas todas fácilmente.

Si después de haber salido del configurador de acceso a redes lo quieres volver a ejecutar alguna vez, el comando es el siguiente:

```
/etc/ppp/configuracion_acceso_a_redes
```

Para conectar se ejecuta:

```
conectar nombre_conexion
```

Si no se especifica ninguna o se da una errónea, se muestra una lista de las conexiones que están configuradas y disponibles.

Para desconectar, basta con ejecutar:

```
desconectar
```

El *script* te preguntará si quieres enviar el correo pendiente antes de terminar la conexión actual.

b) *Scripts* de castellanización: Hay scripts para adaptar el teclado al español, para varios programas, y para la **bash**. Están en **/scripts/castellano**.

c) *Scripts* de instalación de los manuales y del KDE en español: Instalan las páginas **man** existentes traducidas al castellano y el KDE, un entorno gráfico fantástico en castellano.

d) Pásate por la web del EURIELEC para encontrar mas *scripts*, mejoras y correcciones. Están en:

http://www.etsit.upm.es/~eurielec/redhat/
http://personal.redestb.es/guille/Linux/

ALGUNOS TRUCOS

Usar el CD-ROM bajo Linux:

Para usar el CD-ROM bajo Linux, es necesario montarlo. Para eso utiliza el comando:

```
mount -t iso9660 /dev/hdb /mnt/cdrom
```

Suponiendo que tu CD-ROM sea **/dev/hdb** (Maestro en el IDE secundario=hdb, Esclavo en el IDE primario= hdc, esclavo en IDE secundario= hdd), tu CD-ROM estará ahora en el directorio **/mnt/cdrom**.

De todas maneras, si ya tienes el CD-ROM metido en el fichero **/etc/fstab** (compruébalo con **cat /etc/fstab**), podrás montarlo simplemente poniendo:

```
mount /mnt/cdrom
```

Instalar paquetes .RPM (Red Hat):

Para instalar un paquete usa el programa **rpm** de la siguiente manera:

```
rpm -i ficheroAinstalar.rpm
```

Para consultar los paquetes instalados:

```
rpm -q -a | less
Para desinstalar un paquete:
rpm -e nombre_paquete
```

Cómo ejecutar programas del directorio actual:

Por defecto el directorio actual ./ no esta incluido en el **PATH**, en la lista de directorios donde se busca un comando al ejecutarlo. Para ejecutar un programa del directorio en el que estamos basta con poner **./nombre_programa**.

Además podemos añadirlo al **PATH** por defecto, lo que suele ser muy cómodo. Para ello basta con añadir ./ a la linea **PATH** del archivo **/etc/profile**.

Cómo descomprimir ficheros .TAR:

Realmente **tar** no es un tipo de compresor, sino que realmente se encarga de concatenar archivos. En Linux se suele

El equipo de trabajo

Esta distribución ha sido realizada por Eurielec, el club de informática de la ETSI de Telecomunicaciones. Los coordinadores de este proyecto han sido:

- Fernando Herrera de las Heras: <Fer7@writeme.com>
- Guillermo Pérez Pérez: <guille@redestb.es>
- Alejandro Martínez: <amartin@eurielec.clubs.etsit.upm.es>
- Diego Matilla: <dmatilla@eurielec.clubs.etsit.upm.es>

* En el directorio lucas se encuentra toda la documentación sobre linux en castellano en varios formatos.
* En el directorio contrib están los programas suplementarios a la Red Hat.
* En el directorio extra aparecen los programas de empresas comerciales que no tienen nada que ver con Red Hat.
* En el directorio scripts se hallan los scripts necesarios para conectarse a Internet y para castellanizar la distribución.
* El directorio LDP contiene documentacion en ingles en formtato HTML

Para cualquier comentario o sugerencia escriban a:

linux@eurielec.clubs.etsit.upm.es

Queremos mostrar nuestro más sincero agradecimiento a:

- Erik Troan: Por su colaboración y ayuda.
- Guido Reina Alcántara: por su ayuda en la traducción y por su apoyo técnico.
- Pedro Ruiz Maldonado: por su ayuda a la traducción y por su disco duro.
- Francisco Pérez Sorrosal: por su ayuda a la traducción.
- Daniel Lebrero Berna: por su grabadora.
- Alberto Picón: por su ayuda profesional a la traducción.
- Juan Céspedes: por su velocidad de respuesta cuando hacía falta.
- Manuel Arenas: por su ZIP.
- Ramón Gordillo: por preocuparse de que no trabajaramos demasiado.
- Juan Ortega: por su colaboración para enviar los CD's
- a todos los miembros de Eurielec.
- y a todos los que deberían aparecer aquí y no aparecen.

usar mucho, ya que si después se comprime el fichero se suelen obtener mejores resultados (por tratarse de un fichero largo) que cuando se comprimen varios más pequeños.

Para descomprimir un fichero con extensión .tar se debe escribir sobre la línea de comandos:

```
tar -xvf nombre_fichero.tar -C directorio_destino
```

x —> indica eXtraer
v —> Verbose (se muestra información detallada)
f —> para indicar que se trata de un fichero (no de la entrada estándar)
C —> directorio de destino. Si se omite se usa el directorio en el que estás. Es peligroso omitirlo si el fichero tar no crea un directorio (te encontrarás con muchos archivos en el directorio) Puedes averiguarlo ejecutando **tar -vft nombre_fichero.tar** y ver si los archivos que contiene se descomprimen en un directorio.

Para descomprimir un fichero **.tar.gz** o **.tgz** (comprimido además con **gunzip**):

```
tar -xvzf nombre_fichero.tar.gz -C directorio_destino
```

z —> indica que está comprimido con **gunzip**, y que el programa debe descomprimirlo primero.

Cómo descomprimir ficheros .GZ:

```
gunzip nombre_fichero.gz
```

Cómo obtener más ayuda acerca de un comando:

Normalmente la mayoría de los comandos de Linux admiten **—help**, **-h** o **-?** como parámetros para obtener la ayuda en línea. No obstante la mayoría de las veces esta ayuda es un tanto críptica. Suele ser mucho mejor llamar al manual, que es mucho más extenso.

La orden es:

```
man comando_o_tema
```

A veces hay varias entradas para la misma palabra en diferentes secciones del manual sobre ese tema.

Para acceder a dichas páginas se puede escribir:

```
man num_seccion comando_o_tema
```

(Las secciones van del 1 al 8)

Más información:

En Internet consultar:

http://www.etsit.upm.es/~eurielec/redhat
http://personal.redestb.es/guille/redhat

Este proyecto no hubiera sido posible sin la colaboración de RedHat y el esforzado trabajo del EURIELEC. Esperamos que esta distribución sea de tu agrado.